Heikkinen

AEDC-TR-89-7

# PEGSUS 3.0 User's Manual

W. E. Dietz and N. E. Suhs
Calspan Corporation/AEDC Operations

August 1989

Final Report for Period January 1, 1988 – June 30, 1989

ARNOLD ENGINEERING DEVELOPMENT CENTER
ARNOLD AIR FORCE BASE, TENNESSEE
AIR FORCE SYSTEMS COMMAND
UNITED STATES AIR FORCE

**APPROVAL STATEMENT**

This report has been reviewed and approved

Mark S. Briski

MARK S. BRISKI, Capt, USAF
Directorate of Technology
Deputy for Operations

Approved for publication:

FOR THE COMMANDER

Keith L. Kushman

KEITH L. KUSHMAN
Technical Director
Directorate of Technology
Deputy for Operations

| REPORT DOCUMENTATION PAGE | Form Approved OMB No. 0704-0188 |
|---|---|

| | | |
|---|---|---|
| 1. AGENCY USE ONLY *(Leave blank)* | 2. REPORT DATE<br>August 1989 | 3. REPORT TYPE AND DATES COVERED<br>Final Report for 1 JAN 88 to 30 JUN 89 |

| | |
|---|---|
| 4. TITLE AND SUBTITLE<br>PEGSUS 3.0 User's Manual | 5. FUNDING NUMBERS<br>PE-65807F and 62602F |
| 6. AUTHOR(S)<br>Dietz, W. E. and Suhs, N. E., Calspan Corporation, AEDC Operations | |
| 7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)<br>Arnold Engineering Devleopment Center/DOT<br>Air Force Systems Command<br>Arnold Air Force Base, TN 37389-5000 | 8. PERFORMING ORGANIZATION REPORT NUMBER<br>AEDC-TR-89-7 |
| 9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)<br>Arnold Engineering Development Center/DO<br>Air Force Systems Command<br>Arnold Air Force Base, TN 37389-5000 | 10. SPONSORING/MONITORING AGENCY REPORT NUMBER |

11. SUPPLEMENTARY NOTES

Available in Defense Technical Information Center (DTIC).

| | |
|---|---|
| 12a. DISTRIBUTION/AVAILABILITY STATEMENT<br>Approved for public release; distribution is unlimited. | 12b. DISTRIBUTION CODE |

13. ABSTRACT *(Maximum 200 words)*

Many aerodynamic configurations are too complex to be represented by a single computational mesh. A multiple-mesh domain decomposition method called chimera has been developed to allow complex aerodynamic configurations to be modeled by a system of individually generated meshes, each representing a component of the overall configuration. The chimera approach requires the use of two computer codes, PEGSUS and XMER3D. PEGSUS calculates the interactions between the meshes and produces a file consisting of interpolation information that is used by the flow solver to obtain a global flow-field solution for the entire mesh system. This report documents the design and use of the latest version of PEGSUS and describes techniques for the proper modeling of complex aerodynamic configurations.

| | | |
|---|---|---|
| 14. SUBJECT TERMS<br>computational fluid dynamics<br>chimera<br>domain decomposition techniques | grid embedding techniques<br>PEGSUS 3.0<br>XMER3D | 15. NUMBER OF PAGES<br>79<br>16. PRICE CODE |

| | | | |
|---|---|---|---|
| 17. SECURITY CLASSIFICATION OF REPORT<br>UNCLASSIFIED | 18. SECURITY CLASSIFICATION OF THIS PAGE<br>UNCLASSIFIED | 19. SECURITY CLASSIFICATION OF ABSTRACT<br>UNCLASSIFIED | 20. LIMITATION OF ABSTRACT<br>Same as Report |

## GENERAL INSTRUCTIONS FOR COMPLETING SF 298

The Report Documentation Page (RDP) is used in announcing and cataloging reports. It is important that this information be consistent with the rest of the report, particularly the cover and title page. Instructions for filling in each block of the form follow. It is important to ***stay within the lines*** to meet ***optical scanning requirements***.

**Block 1.** Agency Use Only *(Leave blank)*.

**Block 2.** Report Date. Full publication date including day, month, and year, if available (e.g. 1 Jan 88). Must cite at least the year.

**Block 3.** Type of Report and Dates Covered. State whether report is interim, final, etc. If applicable, enter inclusive report dates (e.g. 10 Jun 87 - 30 Jun 88).

**Block 4.** Title and Subtitle. A title is taken from the part of the report that provides the most meaningful and complete information. When a report is prepared in more than one volume, repeat the primary title, add volume number, and include subtitle for the specific volume. On classified documents enter the title classification in parentheses.

**Block 5.** Funding Numbers. To include contract and grant numbers; may include program element number(s), project number(s), task number(s), and work unit number(s). Use the following labels:

| | | | |
|---|---|---|---|
| C | - Contract | PR | - Project |
| G | - Grant | TA | - Task |
| PE | - Program Element | WU | - Work Unit Accession No. |

**Block 6.** Author(s). Name(s) of person(s) responsible for writing the report, performing the research, or credited with the content of the report. If editor or compiler, this should follow the name(s).

**Block 7.** Performing Organization Name(s) and Address(es). Self-explanatory.

**Block 8.** Performing Organization Report Number. Enter the unique alphanumeric report number(s) assigned by the organization performing the report.

**Block 9.** Sponsoring/Monitoring Agency Name(s) and Address(es). Self-explanatory.

**Block 10.** Sponsoring/Monitoring Agency Report Number. *(If known)*

**Block 11.** Supplementary Notes. Enter information not included elsewhere such as: Prepared in cooperation with...; Trans. of...; To be published in.... When a report is revised, include a statement whether the new report supersedes or supplements the older report.

**Block 12a.** Distribution/Availability Statement. Denotes public availability or limitations. Cite any availability to the public. Enter additional limitations or special markings in all capitals (e.g. NOFORN, REL, ITAR).

| | |
|---|---|
| DOD - | See DoDD 5230.24, "Distribution Statements on Technical Documents." |
| DOE - | See authorities. |
| NASA - | See Handbook NHB 2200.2. |
| NTIS - | Leave blank. |

**Block 12b.** Distribution Code.

| | |
|---|---|
| DOD - | Leave blank. |
| DOE - | Enter DOE distribution categories from the Standard Distribution for Unclassified Scientific and Technical Reports. |
| NASA - | Leave blank. |
| NTIS - | Leave blank. |

**Block 13.** Abstract. Include a brief *(Maximum 200 words)* factual summary of the most significant information contained in the report.

**Block 14.** Subject Terms. Keywords or phrases identifying major subjects in the report.

**Block 15.** Number of Pages. Enter the total number of pages.

**Block 16.** Price Code. Enter appropriate price code *(NTIS only)*.

**Blocks 17. - 19.** Security Classifications. Self-explanatory. Enter U.S Security Classification in accordance with U.S. Security Regulations (i.e., UNCLASSIFIED). If form contains classified information, stamp classification on the top and bottom of the page.

**Block 20.** Limitation of Abstract. This block must be completed to assign a limitation to the abstract. Enter either UL (unlimited) or SAR (same as report). An entry in this block is necessary if the abstract is to be limited. If blank, the abstract is assumed to be unlimited.

## PREFACE

The work reported herein was conducted by the Arnold Engineering Development Center (AEDC), Air Force Systems Command (AFSC), under AEDC Project Number DC71PW, at the request of the AEDC/DOT. The AEDC/DOT Project Managers were Dr. K. L. Kushman and Capt. Mark Briski. The results were obtained by Calspan Corporation, AEDC Operations, operating contractor for the aerospace flight dynamics testing facilities at the AEDC, AFSC, Arnold Air Force Base, Tennessee. The work was conducted during the period January 1, 1988 through June 30, 1989, and the manuscript was submitted for publication July 17, 1989.

The authors acknowledge the contributions of Dr. J. C. Erickson, Jr., for his editiorial assistance during the preparation of this report.

## CONTENTS

## ILLUSTRATIONS

# 1.0 INTRODUCTION

Computational fluid dynamics (CFD) calculations in support of ground testing are an important aspect of the AEDC mission. Aerodynamic problems related to wind tunnel testing present two important challenges to CFD, (1) the requirement to produce timely solutions in response to testing requirements, and (2) the solution of problems related to complex three-dimensional configurations. The main obstacle to overcoming these challenges is the difficulty of generating computational meshes. Mesh generation is time-consuming even for relatively simple three-dimensional configurations; for the configurations usually encountered in tunnel testing, generating a suitable mesh may be impractical or topologically impossbile. In addition, a single mesh that is fine enough to resolve the desired aerodynamic features of a complex flow field may be too large for available computer memory.

To address these problems, Benek et al. (Refs. 1, 2, 3, and 4) have developed a method of domain decomposition called "chimera," which allows a system of relatively simple grids, each describing a component of a complex aerodynamic configuration, to be combined into a composite grid to yield solutions to complex flow fields. The chimera scheme has greatly increased the range of aerodynamic problems that can be practically addressed by CFD, and has been applied to store separation problems (Ref. 5), cavity flows (Ref. 6), and transonic tunnel wall interference calculations (Ref. 7).

The general concept behind chimera is illustrated in Fig. 1, which depicts two independently generated meshes modeling a flapped airfoil. The flap mesh is embedded within the airfoil mesh. Clearly, the flap mesh outer boundary can receive flow-field information interpolated from appropriate mesh elements (often referred to as interpolation stencils) of the airfoil mesh. However, a reverse process must occur as well; the airfoil mesh must receive flow-field information from the flap mesh. Since the airfoil mesh has no boundary through which flow-field information can be obtained from the flap mesh, an artificial boundary must be defined within the airfoil mesh. Those mesh points on the artificial boundary that are defined within the airfoil mesh and that are fully contained within the computational region of the flap mesh, can be updated by interpolation from the appropriate mesh elements of the flap mesh. Generally, any mesh can receive information from other meshes through outer boundary and artificial boundary points.

The interpolation process is further illustrated in Fig. 2, which depicts a portion of the overlap region between the airfoil and flap. Airfoil mesh points that are within a certain region surrounding the flap are excluded from the computational domain of the airfoil mesh. (In chimera terminology, they are "blanked" points, or hole points.) The exclusion of points is accomplished by defining a hole creation boundary within the flap mesh that will define a region within which all airfoil points are to be blanked. The points in the airfoil mesh surrounding the blanked points are hole boundary points that receive flow-field information interpolated from mesh elements within the flap mesh, while points on the outer boundary of the flap mesh receive flow-field information interpolated from mesh elements within the airfoil mesh.

Application of the chimera scheme requires two main steps, (1) a description of how each mesh is to communicate flow-field information to other meshes, and (2) execution of a flow solver that uses the communication information generated in the previous step. Presently, two computer programs, PEGSUS and XMER3D, perform Steps 1 and 2, respectively. The processes accomplished by PEGSUS include establishing which boundary points in a mesh can be updated by interpolated flow variables from other meshes, and calculating the required interpolation coefficients for each mesh element sending information to a boundary point. The relation of PEGSUS to the chimera domain decomposition scheme is depicted schematically in Fig. 3. The individual meshes and user-defined mesh connection data are input into PEGSUS, which produces (1) a composite mesh, i.e., a single file consisting of the concatenation of all meshes in the multiple-mesh configuration, and (2) an interpolation file, which is a table associating all boundary points in the composite mesh with mesh elements that supply the boundary points with interpolated flow-field parameters. The composite mesh and interpolation file are input into the XMER3D flow solver, which calculates the flow field in the composite mesh configuration.

The chimera method has evolved in terms of its capabilities and applicability to complex flow configurations. Much of the evolution has been manifested in several versions of the PEGSUS program. PEGSUS 1.x required the definition of a hierarchical mesh structure, which restricted which meshes could embed solid surfaces in other meshes. PEGSUS 2.x allowed much more generality in relationships between meshes by eliminating any hierarchical relationships. However, PEGSUS 2.x was limited in the generality in which mesh-to-mesh communications could be defined, was restricted to simple hole boundary shapes, and required different mesh topologies to be treated as special cases. PEGSUS 3.0 is the latest version of the PEGSUS series of mesh interpolation codes. PEGSUS 3.0 has several enhanced features that increase flexibility and applicability to multiple mesh problems. Most importantly, PEGSUS 3.0 provides greatly increased flexibility in the definition of hole boundaries and mesh interactions. Complex mesh interactions and hole boundary definitions may be handled through input, usually without requiring program modification. Those desiring a more complete description of the chimera concept and the PEGSUS and XMER3D implementations are referred to Refs. 2 and 3. A glossary of PEGSUS terminology is provided in Appendix A. Appendixes B through E contain information about PEGSUS global variables, program structure, subroutines, and I/O files. Appendix F contains sample CRAY® JCL for a PEGSUS application. Appendix G describes the structure of the interpolation file.

# 2.0 GENERAL APPROACH

The purpose of PEGSUS 3.0 is identical to that of earlier versions, namely to produce a composite mesh and interpolation file for the XMER3D flow solver. PEGSUS 3.0 produces files that are identical in format to files produced by earlier versions. The input and output to PEGSUS 3.0 is depicted in Fig. 4. The format of the input mesh files is almost identical to that used in earlier versions; the only difference is a 40-character mesh name that comprises the first record of the mesh input (See Sec. 3.5). Mesh connection information is supplied in NAMELIST format and differs greatly from that used in earlier versions. The output of PEGSUS 3.0, in addition to the composite mesh and interpolation file, consists of (1) a summary of connection and run-time information, (2) diagnostic maps that graphically depict mesh connections, and (3) an information file, which lists every boundary point and its corresponding interpolation element in the composite mesh. The diagnostic maps and information file are described in more detail in Sec. 4.1.

There are four main functional units to PEGSUS 3.0, input, hole location, hole boundary point interpolation, and outer boundary point interpolation. The four functions are performed in the order indicated, and only after all previous functions have been completed. This modularity allows PEGSUS to be executed incrementally, i.e., the functions of hole and outer boundary point interpolation can be performed by "restarting" from output generated from previous functions. Figure 5 is a table listing the seven different modes in which PEGSUS can be run.

## 2.1 INPUT STRUCTURE: MESHES, BOUNDARIES, AND SURFACES

Mesh connection information in PEGSUS 3.0 is defined by three types of relations, mesh-mesh, mesh-boundary, and surface-boundary. The type of information that is supplied in the NAMELIST to define mesh-mesh connections is depicted in Fig. 6. Each mesh in the composite mesh receives information from other meshes in one of two ways, either through its outer boundary or through hole boundaries that have been created by hole creation boundaries in other meshes. In the NAMELIST input, hole boundary and outer boundary linkages are referred to as HBLINKs and OBLINKs, respectively. These linkages are priority lists that indicate the order in which other meshes will be searched for interpolation elements. Figure 6 indicates that Mesh 1 will search Mesh 2, Mesh 3, and Mesh N, in that order, for interpolation elements that can provide information to Mesh 1's hole boundary points. Mesh 2 and Mesh 5 will be searched in a similar manner for interpolation elements that can provide information to the outer boundary of Mesh 1, whereas Mesh 4 will not be searched at all. Each mesh in the composite mesh will be related to the other meshes by similar linkages. PEGSUS 3.0 allows up to 30 outer boundary and 30 hole boundary linkages to be defined for each mesh.

The relationship between meshes and boundaries is depicted in Fig. 7. Boundaries are defined as collections of level surfaces within meshes. OB 1 is the outer boundary of Mesh 1, and HB 1 and HB 2 are hole creation boundaries defined within Mesh 1. Since OB 1 is defined in terms of surfaces within Mesh 1, then in NAMELIST nomenclature, OB 1 "ISPARTOF" Mesh 1. The same "ISPARTOF" relation holds between the hole creation boundaries and Mesh 1. Hole creation boundaries also have relationships to other meshes, i.e., they cause holes in other meshes. This relationship is denoted as "MHOLEIN" (Makes a HOLE IN) in the NAMELIST input. According to Fig. 7, HB 2 makes holes in both Mesh 2 and Mesh 3, and HB 1 makes a hole in Mesh N. Meshes 4 and 5 are not affected by hole creation boundaries defined within Mesh 1. PEGSUS 3.0 allows each mesh to create holes in up to 30 other meshes.

Finally, Fig. 8 illustrates the relation between surfaces and boundaries. Boundaries are comprised of collections of surfaces, and therefore, surfaces have "ISPARTOF" relationships to boundaries in much the same way as boundaries do to meshes. There is no limit in PEGSUS 3.0 to the number of surfaces that may comprise a boundary. As a result, definitions for outer boundaries and hole creation boundaries may be much more complex in PEGSUS 3.0 than would be possible in earlier versions of PEGSUS.

## 2.2 HOLE LOCATION AND FRINGE POINT GENERATION

Identifying hole points requires determining whether a mesh point is inside or outside a hole creation boundary defined in another mesh. A mesh point is considered to be inside a hole creation boundary if it is inside all surfaces that define the boundary. The method used to determine whether a point is inside or outside a surface is illustrated in Fig. 9. A mesh point is considered to be inside a surface if the dot product between $\vec{R}$ (the vector from the closest point on the surface to the mesh point), and $\vec{N}$ (the normal vector on the surface at the closest point, directed outward from the hole region) is negative or zero. If the dot product is positive, the mesh point is considered to be outside the surface.

The hole location process for a hole creation boundary consisting of the three surfaces S1, S2, and S3 is illustrated in Fig. 10. PEGSUS first puts the indices of all points of the mesh in which the hole is to be created into a list. First, all points that are outside the first surface are eliminated from the index list. Then all points contained in the shortened list that are outside the second surface are eliminated, shortening the index list further. When the process is repeated for the third surface, the points remaining in the list are the hole points of the mesh in which the hole is created.

This method of finding hole points has one important restriction, which is illustrated in Fig. 11. In Fig. 11a, a hole creation boundary is defined. The candidate boundary point will be considered to be outside Surface 1, and therefore will be considered to be outside the entire hole boundary. As a result, the point will not be identified as a hole point. Generally, hole creation boundaries (as viewed from the outside) must be convex to obtain correct results. However, PEGSUS 3.0 allows multiple holes to be defined within a mesh. Any hole creation boundary that contains concavities can be divided into one or more entirely convex regions, each of which can create holes in one or more other meshes. If the hole creation boundary is redefined as two separate convex boundaries, as in Fig. 11b, the candidate point will be inside one of the hole creation boundaries, and will therefore be correctly identified as a hole point. The candidate points within the two boundaries will all be labeled as hole points and will therefore effectively merge into a single hole.

Once all holes in all meshes are correctly located, mesh points surrounding holes (i.e., hole boundary points, also called fringe points in chimera terminology) must be identified. The identification of hole boundary points is illustrated in Fig. 12. Hole boundary points are by definition always adjacent to hole points. In PEGSUS 3.0, each hole point has six adjacent points. (In PEGSUS nomenclature, a hole point and its adjacent points comprise a seven-point stencil.) Any adjacent point that is not itself a hole point is identified as a hole boundary point. The pattern of hole boundary points produced by PEGSUS 3.0 will be different from that generated by earlier versions of PEGSUS, which used a 27-point stencil for identifying hole boundary points. This difference is illustrated in Fig. 12, where the hole boundary created by PEGSUS 3.0 is compared to that created by a PEGSUS 2.x version. The PEGSUS 3.0 hole boundary will contain fewer points than the 2.x boundary, resulting in some storage and speed advantage when the XMER3D flow solver is invoked. In addition, the PEGSUS 3.0 hole boundary will be generally closer to the corresponding hole creation boundary than will the corresponding hole boundary created by earlier versions of PEGSUS.

## 2.3 HOLE BOUNDARY INTERPOLATION

Hole boundary interpolation is performed after all hole points and hole boundary (i.e. fringe) points in the composite mesh have been identified. At this point PEGSUS searches other meshes in the composite mesh for mesh elements that may be used to interpolate flow-field information to the hole boundary points.

### 2.3.1 Hole Boundary Priority List

The HBLINK definitions for each mesh (Sec. 2.1) comprise a priority list containing the names of other meshes that are to be searched for interpolation elements that can provide information for hole boundary points. The meshes in the priority list are searched in the order in which they appear in the input. If an interpolation element is found, the interpolation information for the point is stored in a set of arrays. If the hole boundary points cannot be interpolated by mesh elements in the first mesh in the priority list, the second mesh is searched for interpolation elements. The process continues until either (1) interpolation elements for all hole boundary points have been found, or (2) the mesh priority list is exhausted. Any hole boundary points that remain after the mesh priority list is exhausted are termed "orphans." Orphaned hole boundary points will be treated as hole points and hence will not be updated by the flow solver.

Since hole location and hole boundary point interpolation occur as two separate functions, a mesh that causes a hole in another mesh does not necessarily have to be the mesh that provides interpolated information to the resultant hole boundary points. This allows meshes to be used for no other function than to generate holes in other meshes; this feature is used extensively in certain instances, such as the modeling of a fuselage and attached wing (Sec. 5.0).

### 2.3.2 Valid Interpolations

Finding a mesh element that can supply interpolated information to a hole boundary point is a necessary, but not sufficient, condition for an interpolation to be considered valid. The difference between valid and invalid interpolations is illustrated in Fig. 13. In the illustration, the outer boundary of a mesh (dotted lines) is to receive interpolated information from another mesh (solid lines). (We will refer to the first mesh as a "recipient" mesh, and the second mesh as the "donor" of interpolated information). An interpolation is valid if (1) an interpolation element can be found in the donor mesh for which none of the corner points of the element are hole or boundary points, or (2) if a boundary point is coincident with a corner of an interpolation element in the donor mesh, and the corner point itself is not a hole or boundary point. The hole boundary point in Fig. 13 comprises a corner of two mesh elements that contain Points A and B; as a result, Points A and B cannot receive interpolated information from the donor mesh. Point C is coincident with a corner point of the donor mesh. Since the corner point is not itself a boundary or hole point, Point C can receive information from the donor mesh. None of the corner points of the mesh element containing Point D is a boundary or hole point; therefore, Point D will be permitted to receive interpolated information from the donor mesh. In Fig. 13, Points A and B will be orphans unless the recipient mesh has an outer boundary link to another mesh that can provide valid interpolation elements. PEGSUS prints a list of orphan points if any are found; orphans are also identified in graphical form (See Sec. 4.1) for convenience.

### 2.3.3 Stencil-Jumping

Identifying mesh elements that can be used for supplying interpolated information to boundary points requires some sort of searching procedure. An exhaustive search of mesh elements will always find a permissible mesh element if one exists; however, exhaustive searches are computationally intensive. PEGSUS uses a heuristic search mechanism that can find an interpolation element in relatively few steps.

The interpolation algorithm used in PEGSUS is a trilinear interpolation scheme, and is fully documented in Ref. 3. The trilinear interpolation algorithm yields interpolation coefficients for a point either inside or outside the element (the latter case is actually and extrapolation). The interpolation coefficients will all be between 0 and 1.0 if the point is inside the element; a point outside the element will cause at least one coefficient to be greater than 1.0. The interpolation element in effect defines a local coordinate system, while the interpolation coefficients define the location of the point in the local system. In PEGSUS, the interpolation coefficients are used to determine the direction (in computational coordinates) in which the search should proceed in order to eventually find an element that surrounds the point. For instance, a given boundary point and a mesh element with its lowest index corner point at J, K, and L may yield interpolation coefficients of 5.2, 7.8, and 2.3. PEGSUS will then move to the mesh element corresponding to the point J + 5, K + 7, and L + 2. The new mesh element will usually be closer to the boundary point. The procedure is repeated until all interpolation coefficients are between 0 and 1.0. Indices of candidate mesh elements are allowed to jump as much as a third of the distance (in computational coordinates) across a mesh. In general, the search will succeed in three or four steps, even if the initial starting point is far from the final interpolation mesh element. Although this searching mechanism can in principle fail in certain cases (e.g., highly curved or distorted meshes), in practice the searching procedure is highly reliable. This search mechanism is referred to in chimera terminology as "stencil-jumping," where "stencil" in this context refers to interpolation elements, and not to the stencils defined in Sec. 2.2, which are used to identify hole boundary points.

## 2.4 OUTER BOUNDARY INTERPOLATION

The interpolation of outer boundary points is accomplished in almost exactly the same way as hole boundary interpolation, except that outer boundary points comprise the set of candidate boundary points, and a different priority list (i.e. the OBLINK list; see Sec. 2.1) from the one used for hole boundary processing is used. The criteria for valid interpolations and the stencil-jumping algorithm are identical to those used for hole boundary processing. In PEGSUS 3.0, only outer boundary surfaces that are to be interpolated are defined in the input. As a result, free-stream outer boundaries do not have to be specified.

Unlike orphaned hole boundary points, orphaned outer boundary points are not necessarily undesirable. For instance, two meshes may have their outer boundaries only partially embedded within each other; if the entire outer boundary of each mesh is designated as a surface to be interpolated, the points on the boundaries outside of any mesh will be designated as orphans.

Orphaned outer boundary points are not blanked out, and can therefore have boundary conditions (usually free stream) imposed by the XMER3D flow solver.

## 3.0 INPUT IMPLEMENTATION

### 3.1 PARAMETERS

The following constants are defined in PARAMETER statements at the beginning of all program modules. The values of the constants must equal or exceed the expected values for a given composite mesh.

| | |
|---|---|
| MLEN | Maximum number of points in a single mesh. |
| MSLEN | Maximum number of points defining a single surface, maximum number of interpolated points in a single mesh, or maximum number of interpolation elements in a single mesh, whichever is largest. |
| MDIM | Number of meshes in the composite mesh. |
| NBDIM | Number of boundaries in the composite mesh. |
| NSDIM | Number of surfaces in the composite mesh. |

### 3.2 PREPROCESSING

All user-defined mesh connections are input in NAMELIST format. Two features have been added to the standard format to increase readability. FORTRAN-style comment statements (i.e. statements with a "C" in Column 1) may be inserted anywhere in the NAMELIST records, as long as no NAMELIST variables occur on the same line as the comments. Also, the NAMELIST delimiter does not have to be positioned in Column 2, but can be placed anywhere in the line. As a result, the NAMELIST records can be indented to increase readability. The preprocessor in PEGSUS 3.0 inputs the user-defined NAMELIST and corrects the format to make it readable by the compiler.

### 3.3 SYNTAX AND CONSISTENCY CHECK

PEGSUS 3.0 checks the entire NAMELIST input for syntax errors; i.e., PEGSUS will not terminate execution when the first error is encountered, but tries to find all of them at once. If no syntax errors are discovered, PEGSUS then checks for consistency of the input. For instance, PEGSUS will check to see that no mesh is trying to interpolate its boundary conditions from itself, make a hole in itself, or to interpolate its boundary conditions from a mesh that has not been input, etc.

## 3.4 NAMELIST GROUP AND RECORD DEFINITIONS

The NAMELIST input is divided into four group names, $OPTION, $MESH, $BOUNDARY, and $SURFACE. The group names and associated records are defined as follows:

**$OPTION**

MODE = 1, input check only.

= 2, input check and hole location.

= 3, input check, hole location, and hole boundary point interpolation.

= 4, input check, hole location, hole boundary point location, outer boundary point interpolation, and XMER3D interpolation file output.

= 5, hole boundary point interpolation only.

= 6, hole boundary point interpolation, outer boundary point interpolation, and XMER3D interpolation file output.

= 7, outer boundary point interpolation and XMER3D interpolation file output.

Modes 1, 2, 3, and 4 produce a composite mesh. Modes 2, 3, and 5 produce restart files. Modes 5, 6, and 7 require restart files as input. The default is Mode 4 (See chart, Fig. 5).

**$MESH**

ALFA, BETA, GAMA—Euler angles (degrees) determining rotation of input meshes. ALFA, BETA, and GAMA are rotations about the Z, Y, and X coordinates, respectively. Default values for ALFA, BETA, and GAMA are 0.0.

HBLINKn—Priority list of meshes that are to be searched to provide interpolations for hole boundary points. HBLINK1...HBLINK30 are currently supported. Each HBLINKn is a string not exceeding 40 characters. No default values are defined.

JHOLE, KHOLE, LHOLE—Range in the mesh that will be searched for hole points created by other meshes. If it is known *a priori* where hole points are expected in a given mesh, restricting the search range will result in faster execution. The default range is the entire mesh.

NAME—Name of mesh to which NAMELIST record refers. A NAME is a string not exceeding 40 characters. No default values are defined.

OBLINKn—Priority list of meshes that are to be searched to provide interpolations for outer boundary points. OBLINK1...OBLINK30 are currently supported. Each OBLINKn is a string not exceeding 40 characters. No default values are defined.

SCALE—Mesh scaling factor. The original mesh coordinates are multiplied by the scaling factor. The default value is 1.0.

XR, YR, ZR—Point (in translated and scaled coordinates) about which input mesh is to be rotated. Default values are 0.0.

X0, Y0, Z0—Mesh translation factors. These factors are added to the original mesh coordinates. Default values are 0.0.

**$BOUNDARY**

MHOLEINn—Meshes in which the named boundary causes holes. MHOLEIN1...MHOLEIN9 and MHOLEI10...MHOLEI30 are currently supported. Each MHOLEINn is a string not exceeding 40 characters. No defaults are defined.

NAME—Name of boundary. A boundary name is a string not exceeding 40 characters. No default is defined.

ISPARTOF—Mesh which contains boundary. ISPARTOF is a string not exceeding 40 characters. No default is defined.

**$SURFACE**

ISPARTOF—Boundary to which surface belongs. ISPARTOF is a string not exceeding 40 characters. No default is defined.

JRANGE, KRANGE, LRANGE—Ranges of indices that define surface. Maximum allowable ranges are:

JRANGE: 1..JMAX
KRANGE: 1..KMAX
LRANGE: 1..LMAX

No defaults are defined.

NVOUT—Direction of normal outward from hole. NVOUT is required only for hole creation boundaries. Allowable values are "+J," "−J," "+K," "−K," "+L," and "−L." Default is "UNDEF," i.e. undefined.

## 3.5 MESH INPUT

The format of the mesh input files is slightly different from that used in earlier versions of PEGSUS. In PEGSUS 3.0, all meshes must have a unique name, no longer than 40 characters. (Spaces are permissible.) The input format uses three records.

| **Record** | **Variables** |
|---|---|
| 1 | Mesh Name |
| 2 | JMAX, KMAX, LMAX |
| 3 | (((X(J,K,L),J=1,JMAX),K=1,KMAX),L=1,LMAX), (((Y(J,K,L),J=1,JMAX),K=1,KMAX),L=1,LMAX), and (((Z(J,K,L),J=1,JMAX),K=1,KMAX),L=1,LMAX) |

where JMAX, KMAX, and LMAX are the maximum mesh indices in the J, K, and L directions, respectively.

## 4.0 EXAMPLES

Three examples of the application of PEGSUS to multiple-mesh problems of increasing complexity are presented in the following sections.

## 4.1 SPHERE IN CARTESIAN MESH

A simple two-mesh problem is depicted in Fig. 14. A sphere (surrounded by a spherical mesh) is embedded in a Cartesian mesh. The outer boundary of the spherical mesh receives data interpolated from the surrounding Cartesian mesh. Since a solid body is embedded in the Cartesian mesh, a region of the Cartesian mesh must be excluded from computation. The exclusion is achieved by a hole creation boundary defined by a set of surfaces within the spherical mesh. To illustrate the flexibility of PEGSUS 3.0, a hole creation boundary is defined that consists of two hemispheres of unequal radii. If the hole creation boundary were to be defined as a collection of three surfaces (the two hemispherical surfaces and the annular surface between the larger and smaller radii), the hole boundary would contain concavities. For reasons described in Section 2.2, defining a hole boundary with concavities will result in an incorrect hole definition in the Cartesian mesh. The correct way to describe the desired hole boundary is as two separate hole boundaries, neither of which contains any concavities. In this case, one hole boundary is defined as the larger hemisphere and an annular surface extending to the sphere surface; the other hole boundary consists of the smaller hemisphere and a corresponding annular surface. The two hole boundaries will merge into a single hole created in the Cartesian mesh.

The annotated input for the problem of Fig. 14 is given in Fig. 15. There are two mesh, three boundary, and five surface definitions required. Maximum index values for the sphere mesh are 21, 21, and 31 in the J, K, and L directions, respectively. The two hole boundaries are each defined by two surfaces; the outer boundary of the sphere is defined by a single surface. No boundaries or surfaces of the Cartesian mesh are defined in the input, since no surface in the Cartesian mesh receives information from the sphere mesh.

When PEGSUS 3.0 is run, diagnostic maps are produced to aid the user in determining whether the mesh communications are being done properly. Figure 16 is a reproduction of a map produced from the input in Fig. 15. The maps are graphical depictions of the computational meshes which show (1) which points are blanked out hole points, (2) which points are boundary points, and which mesh updates the boundary points, (3) which points are stencil reference points, and which meshes the interpolation stencils update, and (4) which points are "orphans," i.e., hole boundary or outer boundary points which, for some reason, cannot be interpolated from other meshes. The map in Fig. 16 clearly shows the location of the hole in the Cartesian mesh and shows that the hole boundary points are being updated by the sphere mesh. Also, the location of stencil reference points that are interpolating information to the outer boundary of the sphere mesh are shown. With a little practice, a user can obtain much useful information from the maps, such as the amount of mesh overlap, and whether interpolation stencils have been found for boundary points. For example, it can be seen from the map depicted in Fig. 16 that the amount of mesh overlap (i.e. the distance from the hole boundary points to outer boundary of the sphere) is marginal near the bottom of the hole boundary.

PEGSUS 3.0 also prints a table of interpolation information, a section of which is reproduced in Fig. 17 for the input of Fig. 15. The table lists the index of each boundary point for which a valid interpolation has been found, the mesh and stencil that supplies the boundary point with flow-field data, and the corresponding interpolation coefficients. (Note that the mesh receiving interpolated information is referred to as the "recipient," while the mesh providing the interpolated information is the "donor.") The number of boundary points and interpolation stencils in each mesh is included at the beginning of the tabular data for each mesh. Information that is necessary for the proper definition of array dimensions in the XMER3D flow solver are also included, such as the pointer variables IISPTR and IIEPTR.

## 4.2 INLET/FOREBODY CONFIGURATION

The second example is an inlet/forebody configuration, shown in Fig. 18. The configuration consists of two meshes. The first mesh consists of the forebody. The second mesh models the inlet interior surfaces and a region ahead of the inlet. The inlet mesh passes through the inlet region of the forebody mesh. The outer boundary of the inlet mesh forward of the inlet face receives flow-field information interpolated from the appropriate mesh elements of the forebody mesh. The surface of the forebody mesh corresponding to the inlet face receives flow-field information interpolated from the inlet mesh. Solid-wall boundary conditions are imposed on the interior walls of the inlet, and mass flow conditions are imposed at the downstream face of the inlet mesh. The input that defines the appropriate mesh communications is reproduced in Fig. 19. The flexibility of the PEGSUS 3.0 input allows any surface to be defined as an outer boundary. This flexibility is exploited in the inlet/forebody configuration. The surface of the forebody in the vicinity of the inlet-forebody juncture is defined as an "outer" boundary, which receives interpolated information from the inlet mesh. It should be noted that no holes are created in either mesh, since no solid surface from one mesh is embedded within the discretized portion of the other mesh.

## 4.3 STORE/STING CONFIGURATION IN A CAVITY

The final example is a complex aerodynamic configuration consisting of a store, sting support, and solid-wall cavity. The store and sting are each modeled by a single mesh (Fig. 20). The meshes overlap at the interface between the store and the sting. Since the solid surfaces of the store and sting meshes do not intrude into the discretized regions of either mesh, the store and sting meshes do not create holes in each other. However, the store and sting meshes do create holes in the cavity meshes, as shown in Fig. 21. The cavity is modeled by four meshes, while the region exterior to the cavity is modeled by two meshes. The outer boundaries of the cavity meshes and the exterior meshes either (1) are set at free-stream conditions (exterior mesh only), (2) have solid-wall boundary conditions imposed (part of the lower boundary of the exterior mesh, and some surfaces of the cavity meshes, or (3) receive information interpolated from neighboring meshes. The store is contained entirely within the cavity, whereas the sting is contained in both the cavity and one of the exterior meshes. The boundaries of the holes receive information interpolated from the interiors of the store and sting meshes, while the outer boundaries of the store and sting meshes receive information interpolated from the cavity and exterior meshes. The complete mesh system is depicted in Fig. 22.

The input for the store/sting/cavity configuration is reproduced in Fig. 23. Note that only outer boundary surfaces that are to be interpolated from other meshes need be included in the outer boundary descriptions; those surfaces that have either free-stream or solid-wall boundary conditions imposed need not be included in the input. Also, note that not all outer boundary and hole boundary linkages (OBLINKn and HBLINKn) defined in the input are necessarily required; however, no penalty is incurred by specifying outer boundary and hole boundary linkages that are not used.

## 5.0 COMMON MODELING PROBLEMS

The casting of a computational aerodynamic problem into a multiple-mesh formulation allows simpler meshes to be used than would be possible in a single-mesh formulation. However, sometimes even the simpler meshes may, because of geometric constraints, exhibit undesirable characteristics such as internal discontinuities or extreme skewness. Because of the nature of the algorithms employed in PEGSUS 3.0, undesirable mesh features may result in erroneous results unless proper care is exercised.

Almost all errors resulting from undesirable mesh features will be manifested as incorrect hole locations. In these cases, a review of the diagnostic maps will reveal hole points occurring in regions that should not be blanked out. For instance, Fig. 24a depicts a hole creation boundary that has been defined as a single level surface within a mesh. In this case, the level surface contains a sharp discontinuity. PEGSUS 3.0 will attempt to identify points belonging to other meshes that fall inside the hole creation boundary. The hole location procedure will, for the candidate point shown, (1) identify the point on the cusp of the level surface as the closest point to the candidate point, (2) calculate the dot product of the normal at the cusp and the vector defined by the cusp and candidate points, and (3) decide the candidate is inside the hole creation boundary, because the dot product found in the previous step will be negative. This determination is clearly incorrect; overall, the effect of the discontinuity in the hole creation boundary will be a "cloud" of hole points surrounding the cusp.

Fortunately, the flexible surface and boundary definitions incorporated into PEGSUS 3.0 allow such problems to be solved by modifying the input. If the hole creation boundary is divided at the discontinuity into two separate surfaces (Fig. 24b), the candidate point will be found to be (1) inside Surface 1, (2) outside Surface 2, and therefore, (3) outside the hole creation boundary. It is good practice in general to construct hole creation boundaries such that each surface comprising the boundary is smooth.

Another common problem occurs in discontinuous regions of meshes. Figure 25a depicts a hole creation boundary defined as a level surface in an O-mesh in which the edges of the boundary are coincident in the region of the cut. Here the closest points to a candidate point are the coincident points on the cut. PEGSUS will simply choose one of the points as the closest to the candidate (i.e., the first one it finds). A problem arises if the point chosen is associated with a normal vector pointing away from the candidate point; in this case the candidate will be determined to be inside the hole creation boundary. Typically this problem manifests itself as a funnel-shaped "cloud" of hole points extending from the cut region of the hole creation boundary into the interior of the mesh containing the candidate point. The solution of this problem is similar to that employed in the previous example. If the hole creation boundary is divided into two surfaces, as shown in Fig. 25b, the candidate point will be found to be outside the hole creation boundary.

Another example is illustrated in Fig. 26a. Two meshes are included in this configuration. The first mesh models a body with an attached fin. The second mesh models a strut attached to the body. The mesh topology of the body/fin mesh is such that the fin and body surfaces lie on level surfaces that are normal to different computational coordinates. It is necessary to blank out points in the strut mesh that lie within the body and fin. Initially, a hole creation boundary is created from the two level surfaces comprising the body and fin. Application of the hole location algorithm will have an undesired effect; all points within the body will be inside Surface 2 and outside Surface 1, whereas all points within the fin will be outside Surface 2 and inside Surface 1. Since a point must be inside all surfaces to be considered inside a boundary comprised of those surfaces, no points will be blanked out.

The solution to this problem is depicted in Fig. 26b, where the original hole creation boundary has been divided into two separate boundaries, each comprised of a single-level surface. Now the points inside the fin will be inside Boundary 1, and the points inside the body will be inside Boundary 2. Note that the fact that the fin points are outside Boundary 2 has no overall effect, since a point is blanked if it is inside any boundary.

Modeling of wing/body junctures is a common requirement in multiple-mesh configurations. Unfortunately, a complex wing/body juncture can result in the incorrect blanking of hole points if the juncture is not modeled correctly. Figure 27a depicts the junction of a wing and a body. The boundary of the wing mesh extending from the wing root is coincident with the surface of the fuselage. As a result, the fuselage surface does not create a hole in the wing mesh. The wing does, however, create a hole in the fuselage mesh. A hole creation boundary defined in the wing mesh cannot be completely closed; an aperture in the hole creation boundary will exist at the intersection of the wing surface and the hole creation boundary. As a result, some points close to the fuselage body may be considered by the hole location algorithm to be outside the hole creation boundary; these points will not be blanked as required.

A solution to this problem is illustrated in Fig. 27b. The structure of PEGSUS 3.0 allows a mesh to do nothing more than create holes in other meshes. A mesh of this type is "fictitious" in that it does not interact in any other way with the other meshes; i.e., it does not provide interpolated information to boundaries of other meshes, nor does it receive information interpolated from other meshes. In Fig. 27b, the hole creation boundary shown in Fig. 27a has been used to define the outer boundary of a fictitious mesh. The fictitious mesh extends into the interior of the fuselage mesh. The outer boundary of the fictitious mesh is defined as a hole creation boundary and, therefore, creates a hole in the fuselage mesh. In effect, the fictitious mesh acts to create a hole in the fuselage mesh for the wing mesh. Since the hole creation boundary of the fictitious mesh is completely closed, hole points are correctly identified in the fuselage mesh.

It should be noted that the surfaces of the fuselage and wing do not create holes in the fictitious mesh, since the fictitious mesh is not used in flow computations. The present version of PEGSUS will include fictitious meshes in the composite mesh file; the associated records should be removed from the composite mesh file before it is used in the flow solver.

The previous examples show how common modeling problems can be treated through modifications to input only. Earlier versions of PEGSUS often required modifications to the program code to alleviate problems caused by mesh discontinuities or skewness. The capability exists within the input structure of PEGSUS 3.0 to define boundaries and surfaces in a manner that can successfully accommodate most problems.

## 6.0 CONCLUDING REMARKS

PEGSUS 3.0, a code used in the chimera multiple-mesh flow-field calculation scheme, has been developed, implemented, and applied successfully to aerodynamic configurations that earlier versions of PEGSUS had been unable to process without extensive code modifications. PEGSUS 3.0 is an improvement over earlier versions in that (1) all structured mesh topologies may be accommodated, (2) more flexibility is allowed for defining hole and outer boundary surfaces, (3) the input has been restructured into a more readable and usable form, and (4) diagnostic aids have been added to assist in the correct formulation of multiple-mesh problems.

## REFERENCES

1. Benek, J. A., Steger, J. L., and Dougherty, F. C. "A Flexible Grid Embedding Technique with Application to the Euler Equations." AIAA Paper No. 83-1944, July 1983.

2. Benek, J. A., Buning, P. G., and Steger, J. L. "A 3-D Chimera Grid Embedding Technique." AIAA Paper No. 85-1523, July 1985.

3. Benek, J. A., Dougherty, F. C., and Buning, P. G. "Chimera: A Grid-Embedding Technique." AEDC-TR-85-64 (AD-A167466) December 1985.

4. Benek, J. A., Donegan, T. L., and Suhs, N. E. "Extended Chimera Grid Embedding Scheme with Applications to Viscous Flows." AIAA Paper No. 87-1126, June 1987.

5. Dougherty, F. C., Benek, J. A., and Steger, J. L. "On Applications of Chimera Grid Scheme to Store Separation." NASA-TM-88193, October 1985.

6. Suhs, N. E. "Computations of Three-Dimensional Cavity Flow at Subsonic and Supersonic Mach Numbers." AIAA Paper No. 87-1208, June 1987.

7. Donegan, T.L., Benek, J.A., and Erickson, J.C. "Calculation of Transonic Wall Interference." AIAA Paper No. 87-1432, June 1987.

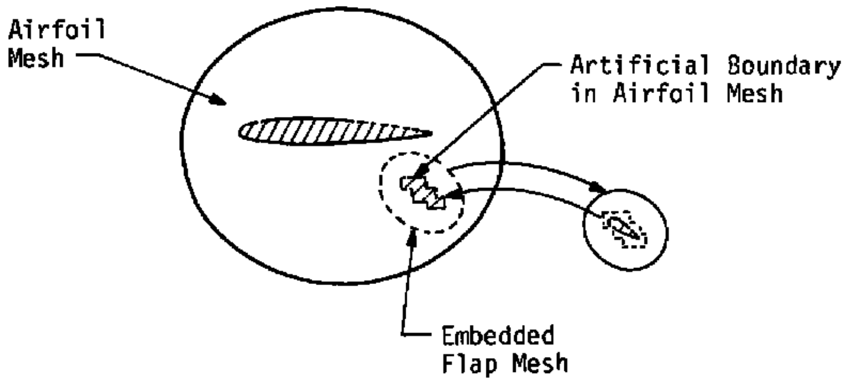


**Figure 1. Mesh-to-mesh communication.**

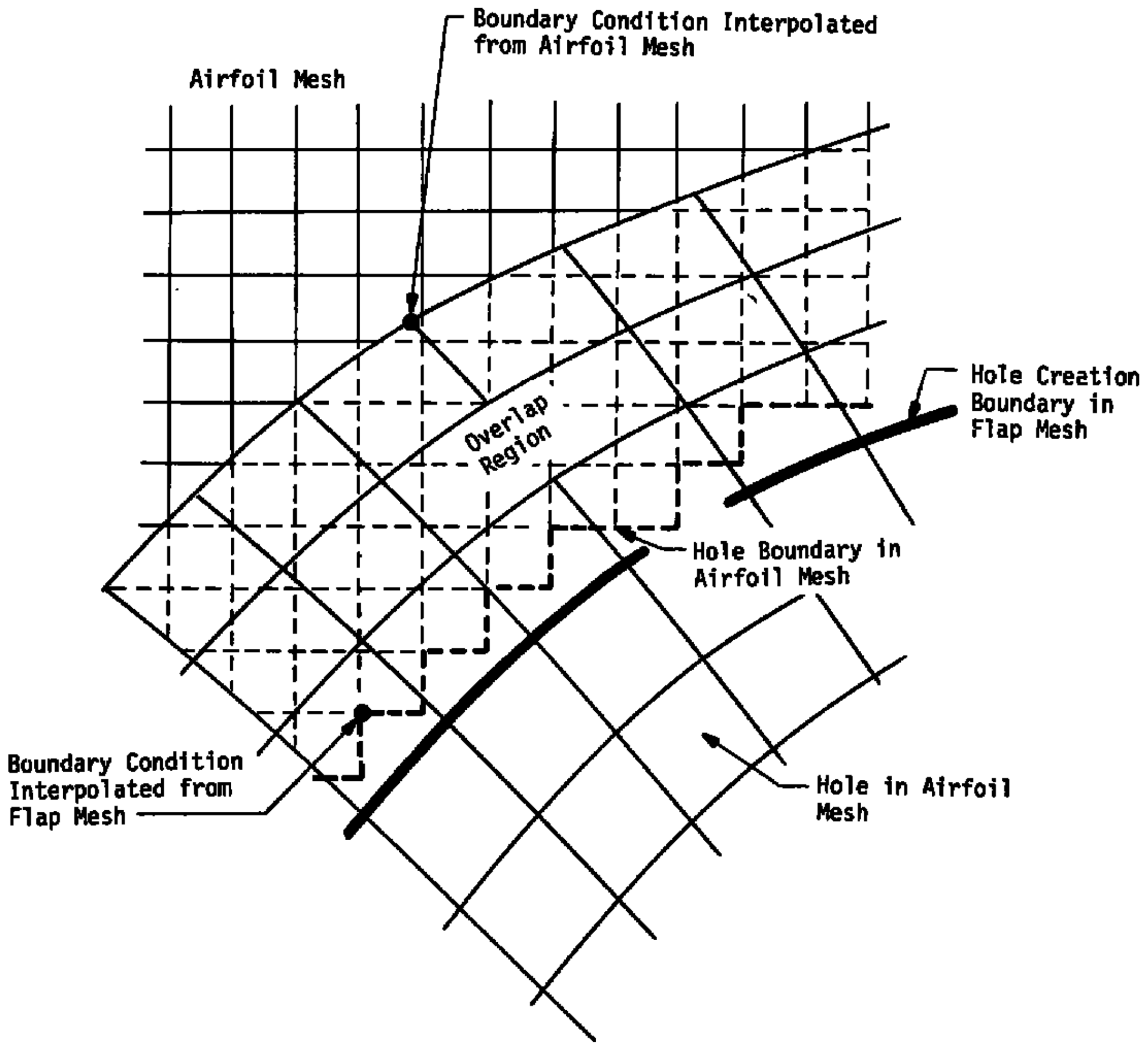

Boundary Condition Interpolated from Airfoil Mesh
Airfoil Mesh
Hole Creation Boundary in Flap Mesh
Overlap Region
Hole Boundary in Airfoil Mesh
Boundary Condition Interpolated from Flap Mesh
Hole in Airfoil Mesh

**Figure 2. Overlap region between grids.**

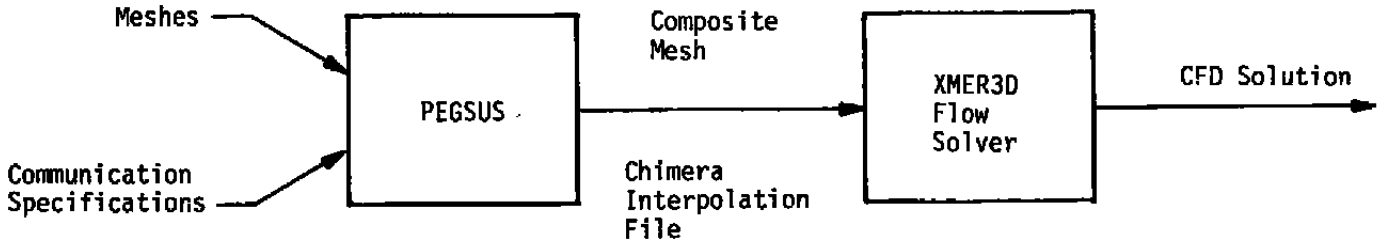


**Figure 3. Chimera scheme.**

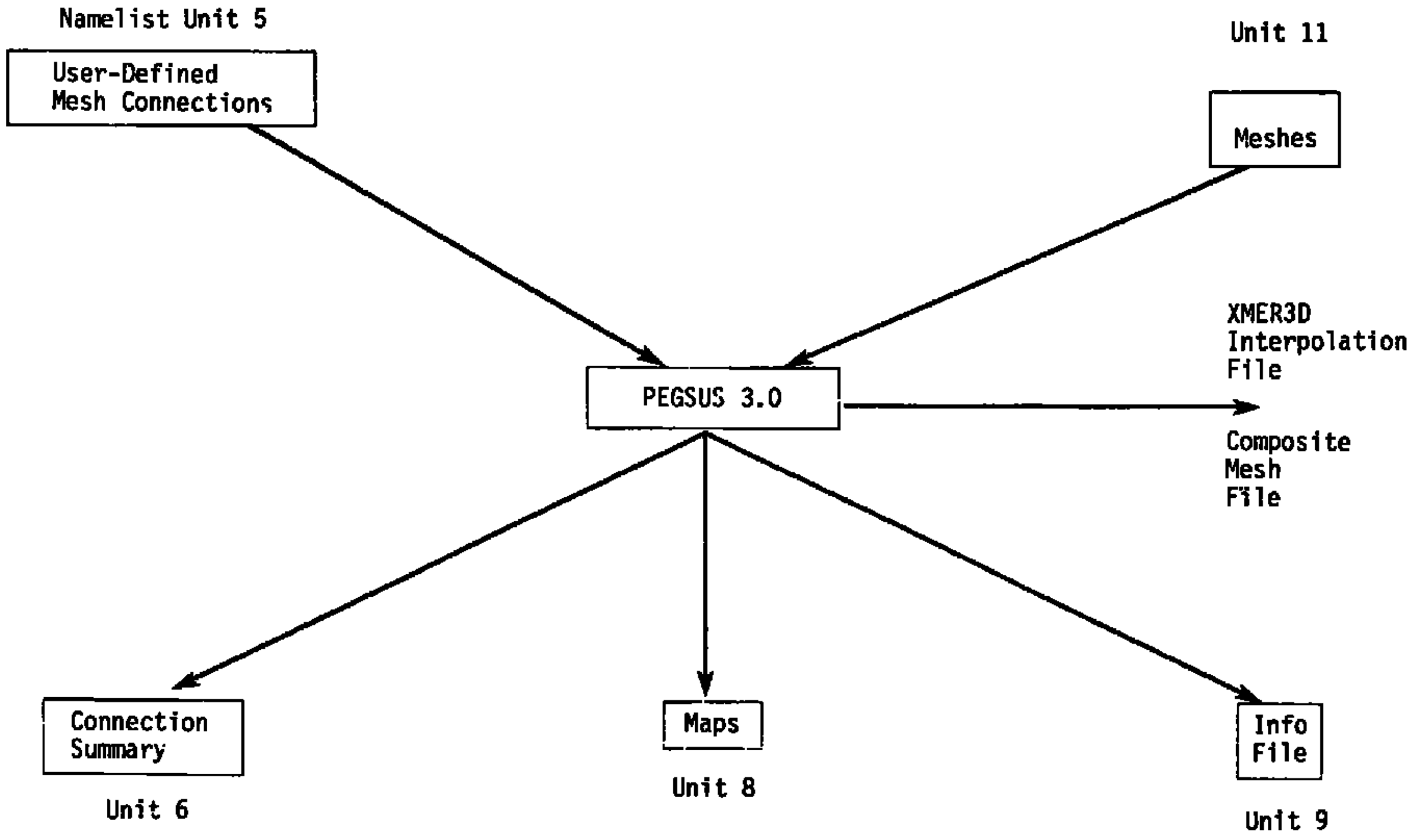


**Figure 4. PEGSUS 3.0 input and output files.**

| | Mode | | | | | | |
|---|---|---|---|---|---|---|---|
| | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
| Input | X | X | X | X | | | |
| Hole Location | | X | X | X | | | |
| Hole Boundary Point Interpolation | | | X | X | X | X | |
| Outer Boundary Point Interpolation | | | | X | | X | X |

**Figure 5. PEGSUS 3.0 modes.**

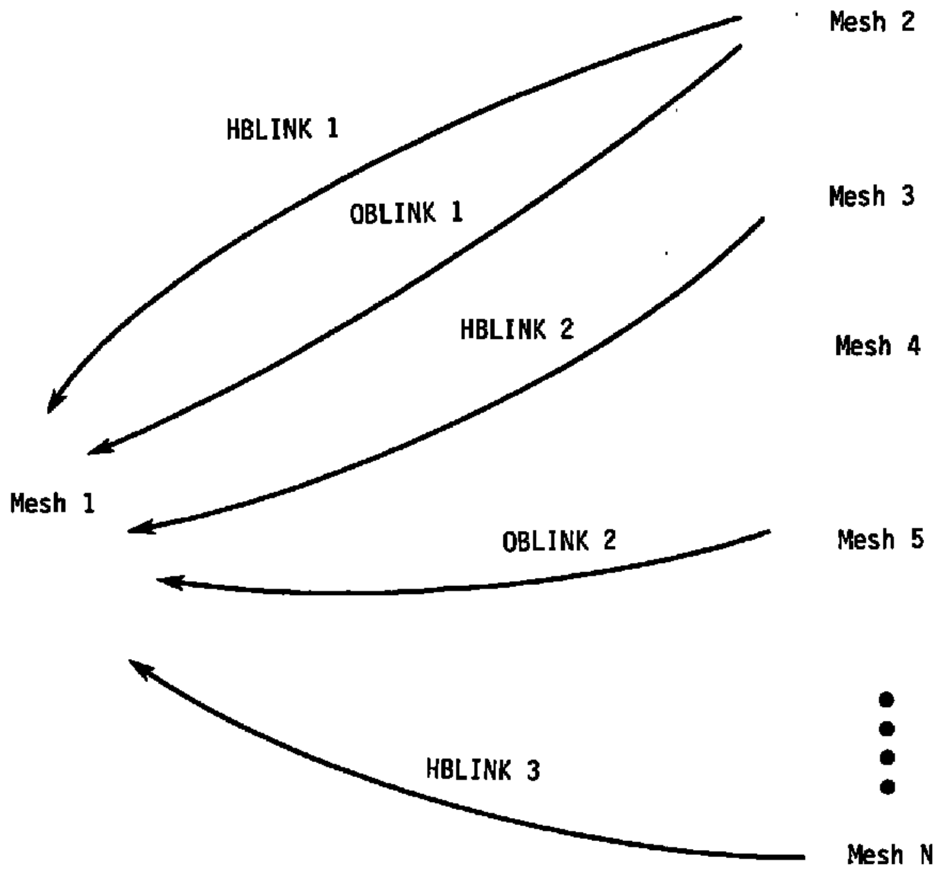


**Figure 6. Mesh-mesh connections.**

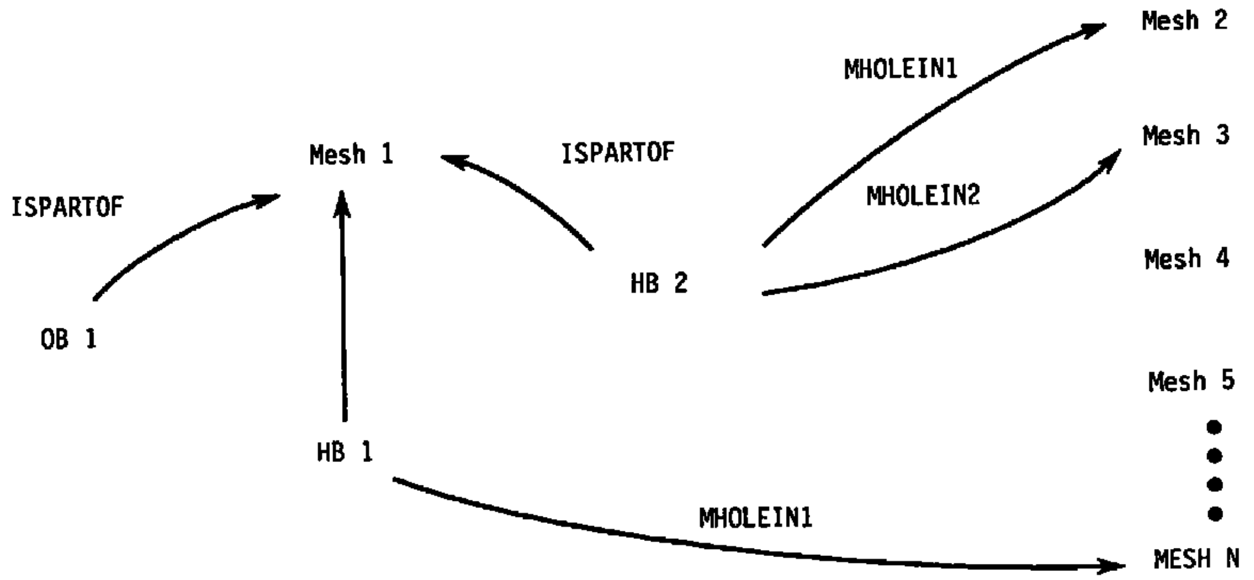


Figure 7. Mesh-boundary relations.

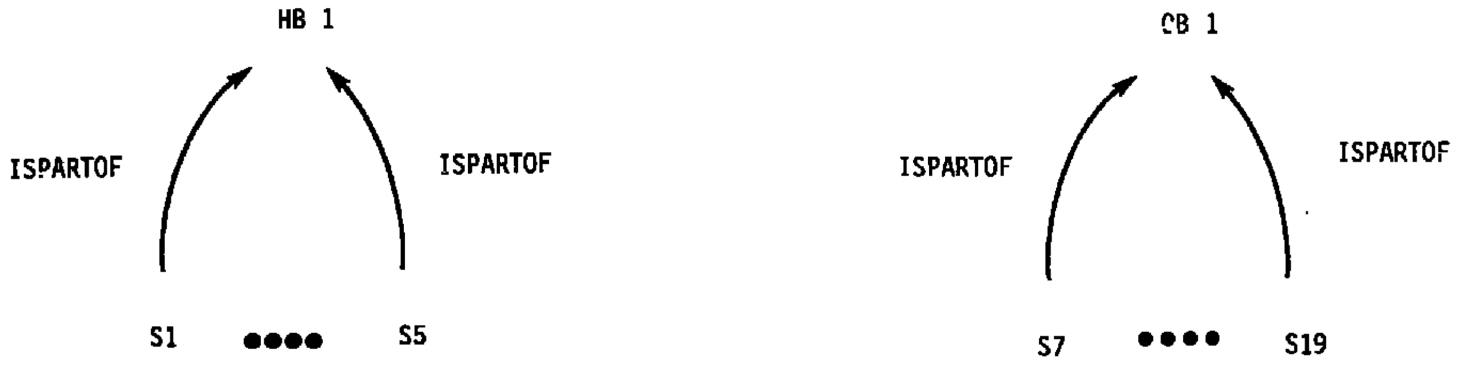


Figure 8. Surface-boundary relations.

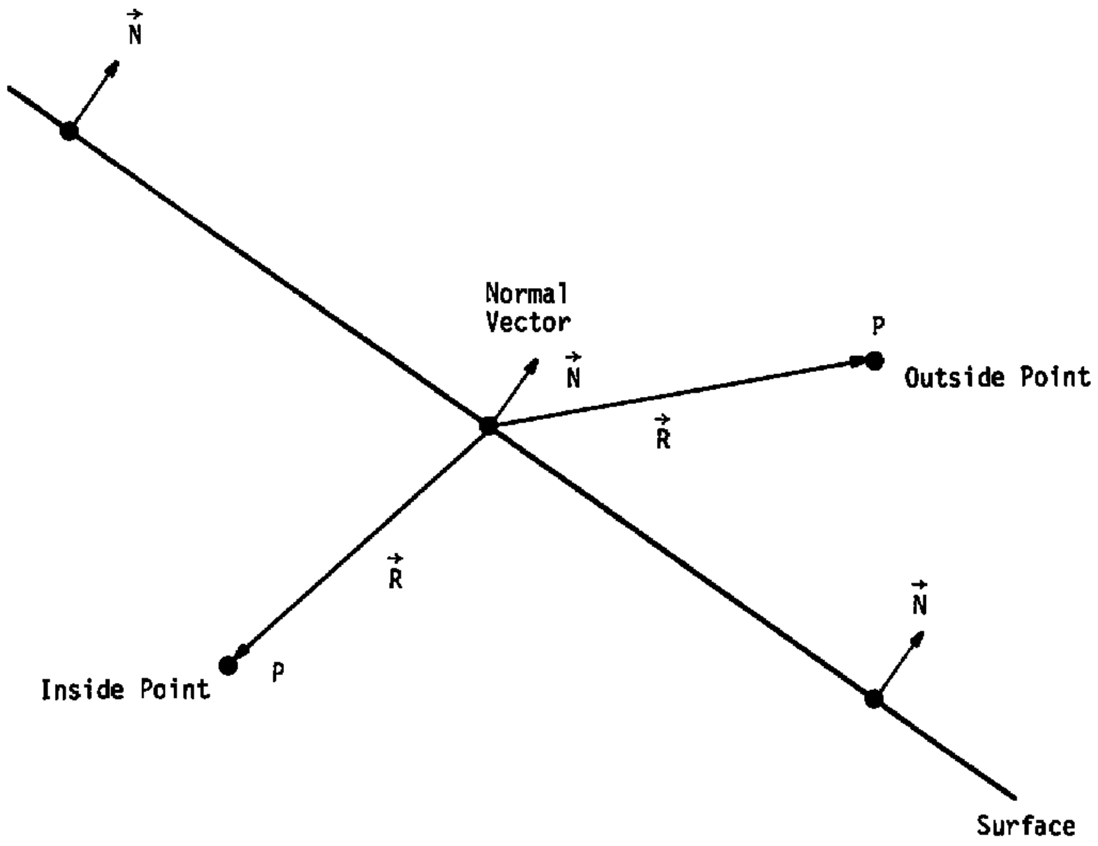


Figure 9. "Inside" and "outside" a surface.

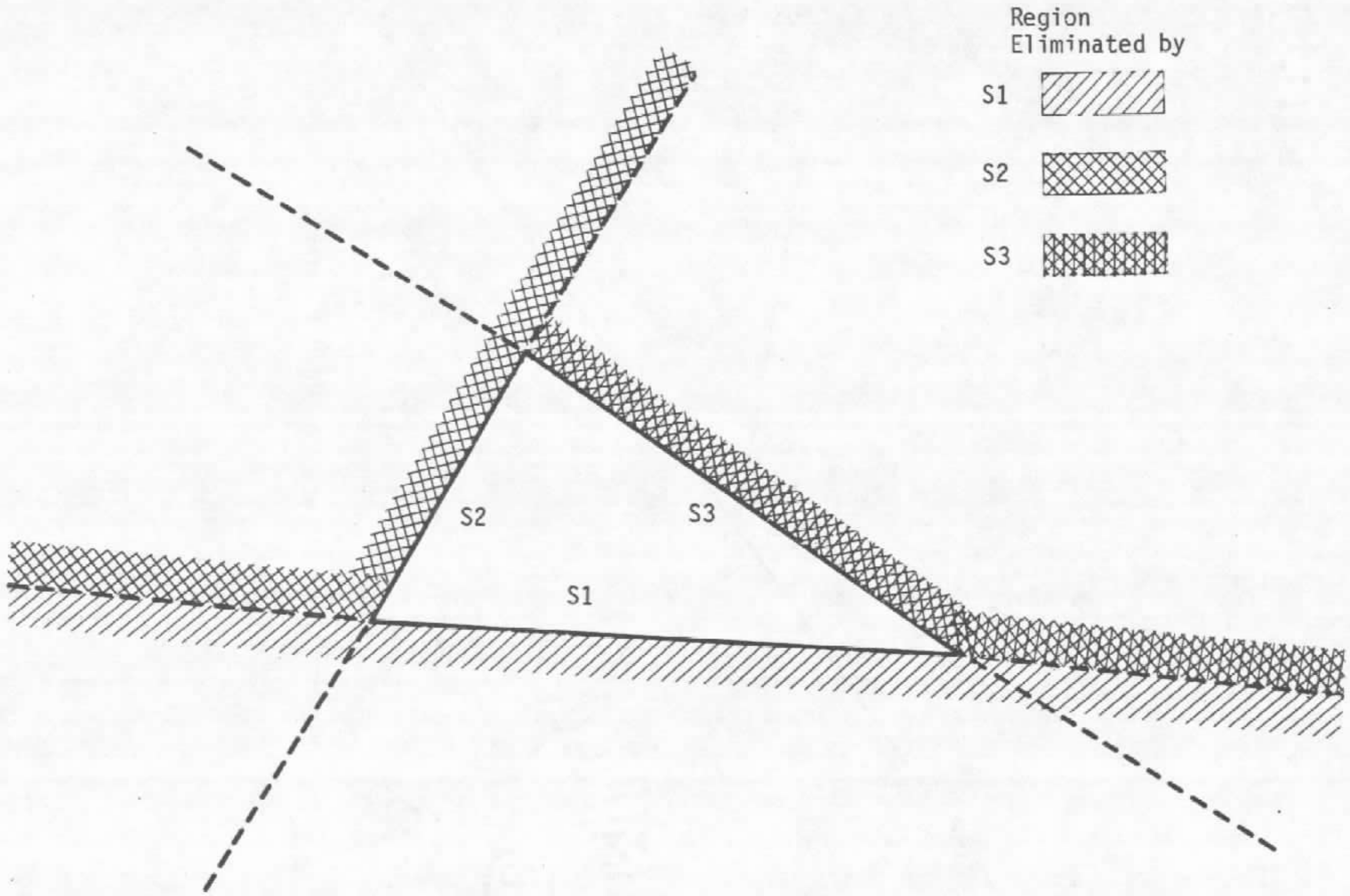
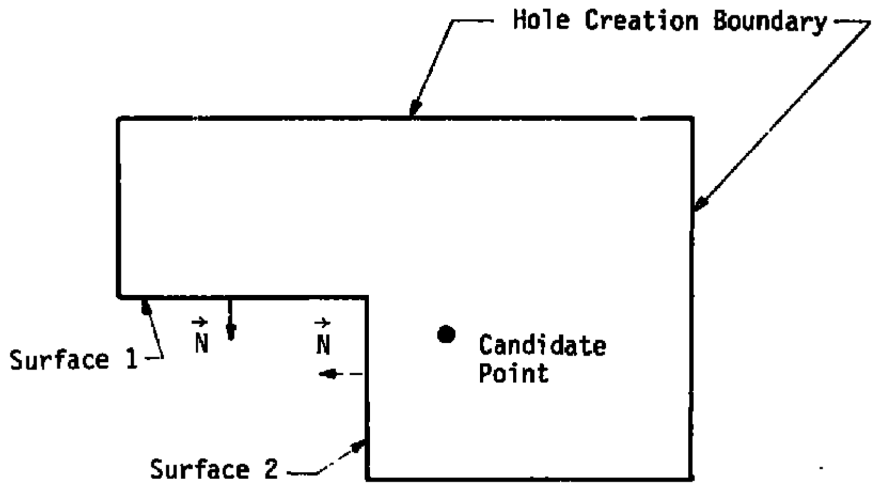
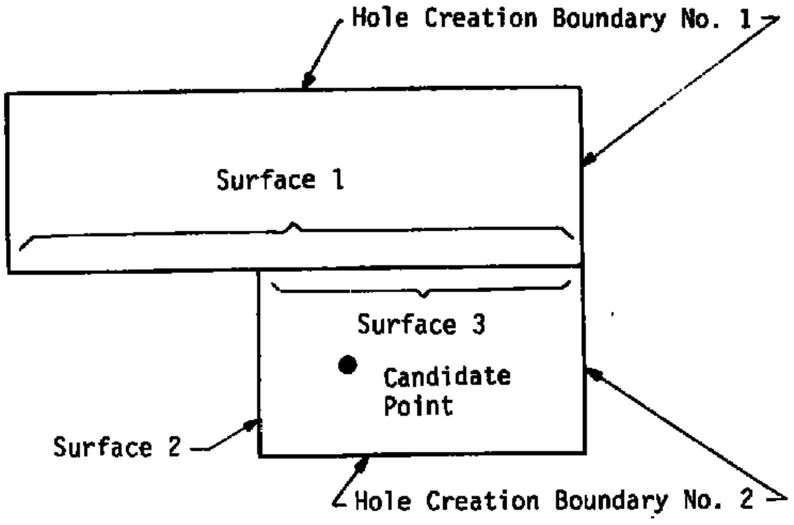


Figure 10. Hole location.



a. Hole creation boundary with concavity



b. Removal of concavity
Figure 11. Restrictions on hole creation boundaries.

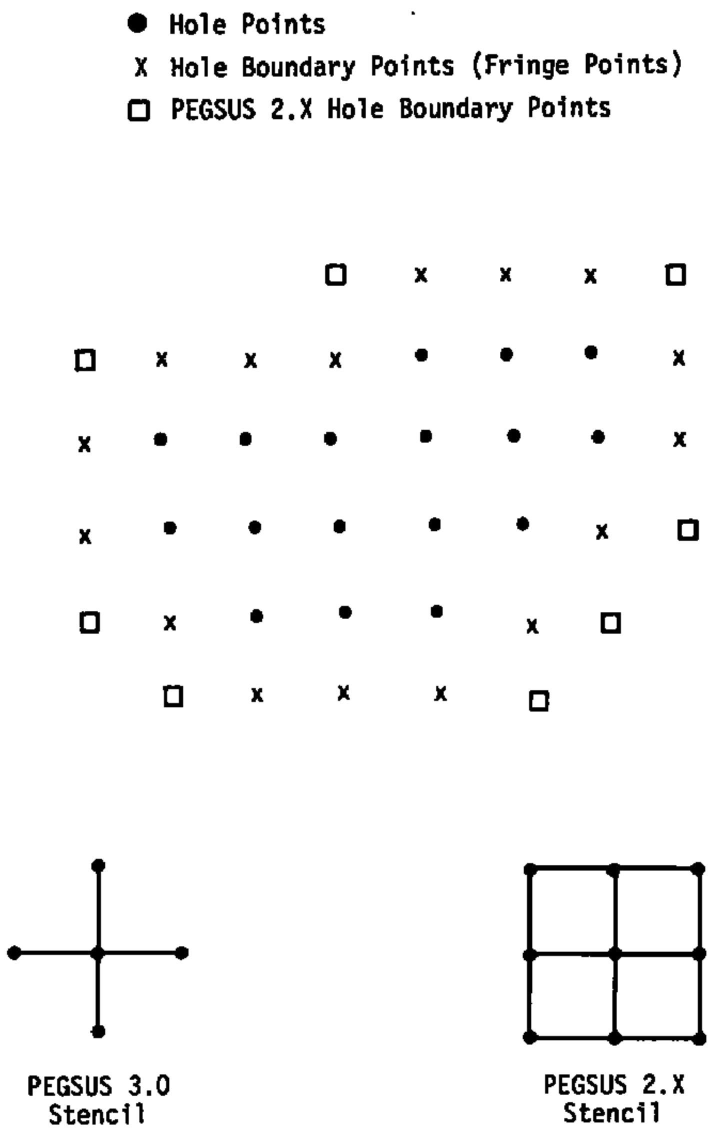

● Hole Points
X Hole Boundary Points (Fringe Points)
□ PEGSUS 2.X Hole Boundary Points
PEGSUS 3.0 Stencil
PEGSUS 2.X Stencil

**Figure 12. Hole boundary generation.**

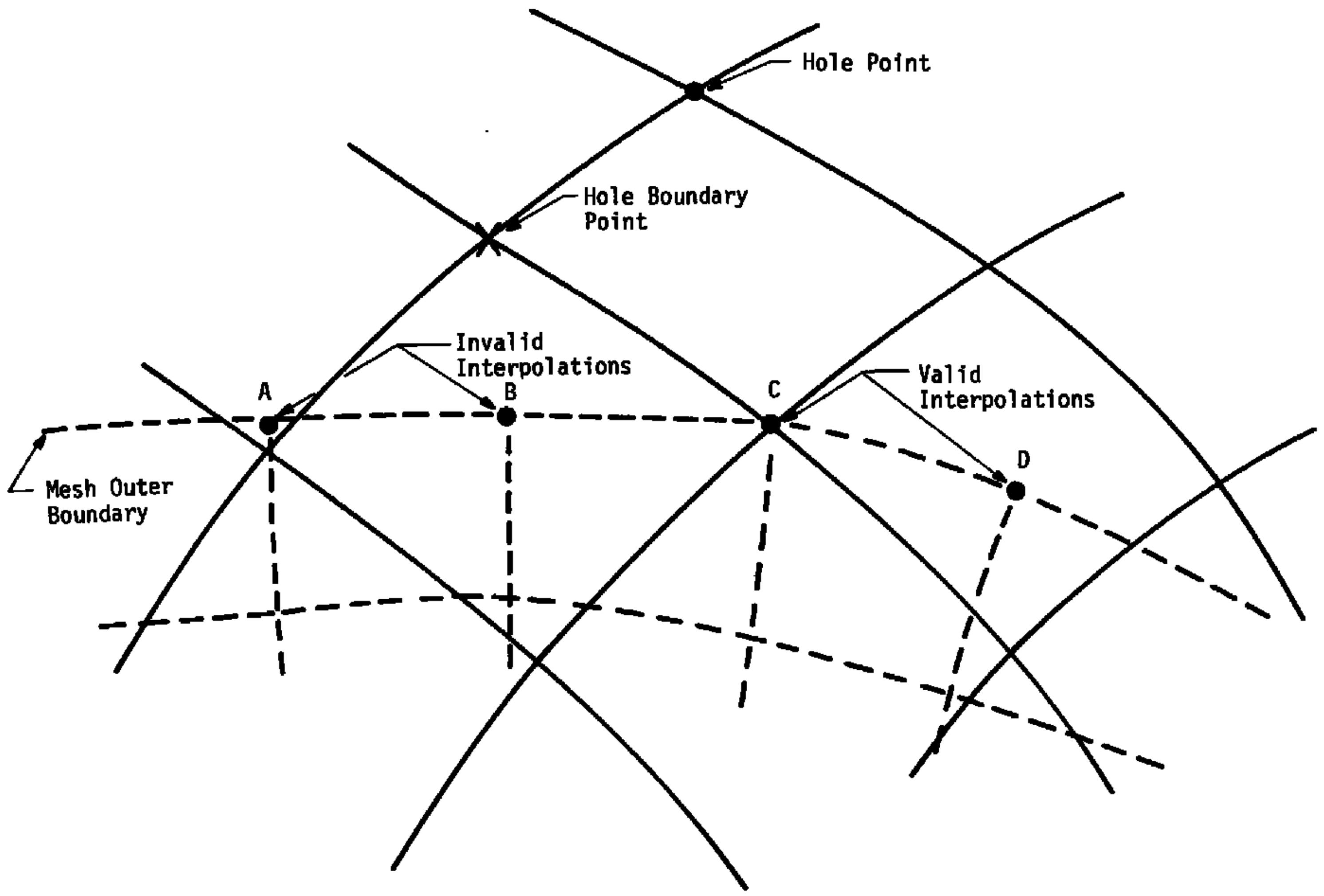


Figure 13. Valid and invalid interpolations.

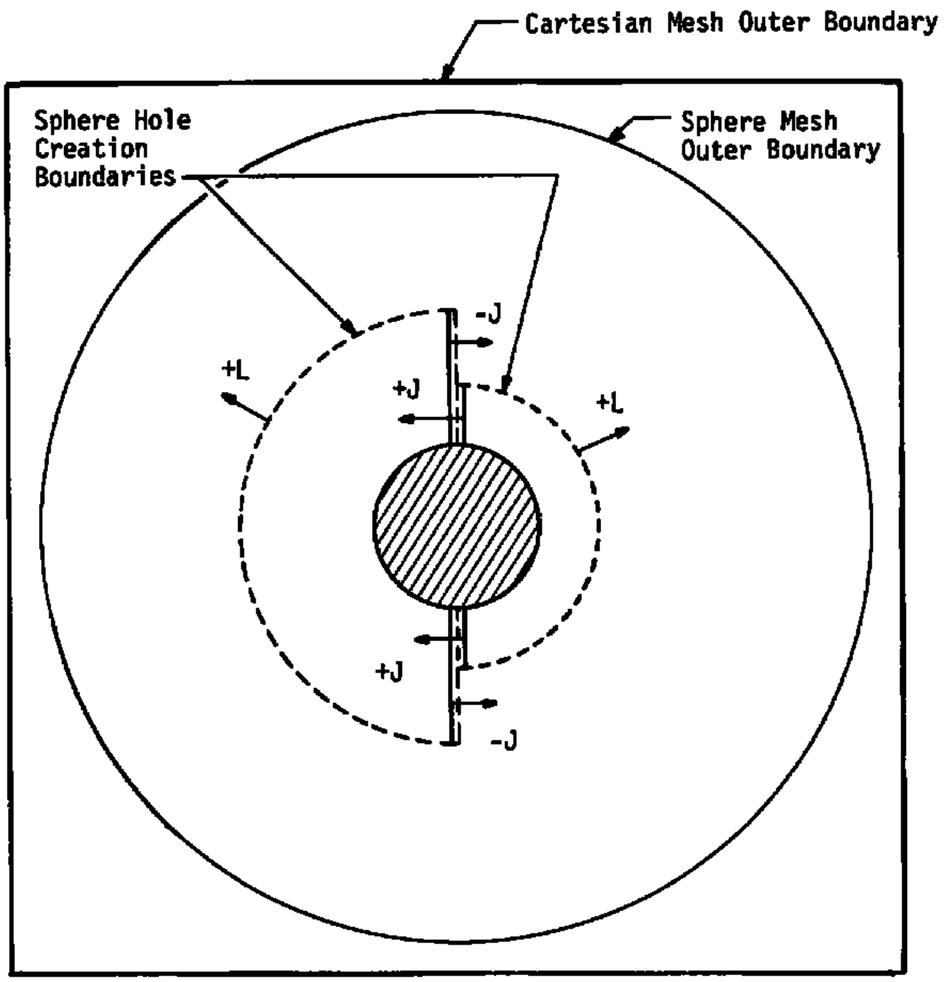

Cartesian Mesh Outer Boundary
Sphere Hole Creation Boundaries
Sphere Mesh Outer Boundary
-J
+J
+L
+L
+J
-J

**Figure 14. Plane of symmetry for sphere embedded in Cartesian mesh.**

```
C                        PEGSUS 3.0 INPUT

C                     CARTESIAN GRID / SPHERE

C  OPTIONS
 $OPTION
C   SPECIFY PEGSUS MODE
    MODE = 4,
              $END


C  MESH DEFINITIONS

 $MESH NAME = 'CARTESIAN GRID',
C       CARTESIAN MESH WILL INTERPOLATE HOLE BOUNDARY POINTS
C       CARTESIAN MESH SIZE: JMAX = 41, KMAX = 21, LMAX = 21
C  FROM SPHERE MESH
      HBLINK1 = 'SPHERE',
                                 $END
 $MESH NAME = 'SPHERE',
C       SPHERE WILL INTERPOLATE ITS OUTER BOUNDARY POINTS
C  FROM THE CARTESIAN MESH
C      SPHERE MESH SIZE: JMAX = 21, KMAX = 21, LMAX = 31
      OBLINK1 = 'CARTESIAN GRID',
      X0 = 5.0 ,Y0 = 5.0, Z0 = 5.0,
                                 $END

C  BOUNDARY DEFINITIONS

 $BOUNDARY  NAME = 'SPHERE HOLE BOUNDARY 1',
C     FIRST HOLE CREATION BOUNDARY MAKES A HOLE IN THE
C  CARTESIAN MESH AND IS DEFINED AS PART OF THE SPHERE MESH
           MHOLEIN1 = 'CARTESIAN GRID',
           ISPARTOF = 'SPHERE',
                                 $END

 $BOUNDARY  NAME = 'SPHERE HOLE BOUNDARY 2',
C     SECOND HOLE CREATION BOUNDARY IS DEFINED IDENTICALLY
C  TO THE FIRST (SURFACES WILL BE DEFINED DIFFERENTLY)
           MHOLEIN1 = 'CARTESIAN GRID',
           ISPARTOF = 'SPHERE',
                                 $END

 $BOUNDARY  NAME = 'SPHERE OUTER BOUNDARY',
C     SPHERE OUTER BOUNDARY IS DEFINED AS PART OF THE SPHERE MESH
           ISPARTOF = 'SPHERE',
                                 $END


C  SURFACE DEFINITIONS

C       THE FIRST SURFACE COMPRISING THE FIRST SPHERE HOLE CREATION
```

**a. Page 1**

**Figure 15. Input for sphere/Cartesian mesh.**

```
C  BOUNDARY IS DEFINED BY THE FOLLOWING SURFACE AND OUTWARD
C  NORMAL DIRECTION
 $SURFACE ISPARTOF = 'SPHERE HOLE BOUNDARY 1',
           JRANGE = 1,11,
           KRANGE = 1,21,
           LRANGE = 6,6,
           NVOUT = '+L',
                                 $END

 $SURFACE ISPARTOF = 'SPHERE HOLE BOUNDARY 1',
           JRANGE = 11,11,
           KRANGE = 1,21,
           LRANGE = 1,6,
           NVOUT = '+J',
                                 $END

C   DEFINITION OF THE SECOND SPHERE HOLE CREATION BOUNDARY
 $SURFACE ISPARTOF = 'SPHERE HOLE BOUNDARY 2',
           JRANGE = 11,21,
           KRANGE = 1,21,
           LRANGE = 21,21,
           NVOUT = '+L',
                                 $END

 $SURFACE ISPARTOF = 'SPHERE HOLE BOUNDARY 2',
           JRANGE = 11,11,
           KRANGE = 1,21,
           LRANGE = 1,21,
           NVOUT = '-J',
                                 $END

 $SURFACE ISPARTOF = 'SPHERE OUTER BOUNDARY',
C      SPHERE OUTER BOUNDARY REQUIRES NO OUTWARD NORMAL DEFINITION
           JRANGE = 1,21,
           KRANGE = 1,21,
           LRANGE = 31,31,
                                 $END

C  END OF SPHERE/CARTESIAN MESH PEGSUS 3.0 INPUT
```

**b. Page 2**
**Figure 15. Concluded.**

```
                    DIAGNOSTIC MAPS
          LEGEND

BOUNDARY POINTS UPDATED BY:
 A  -  CARTESIAN GRID
 B  -  SPHERE


 INTERPOLATION STENCILS UPDATING POINTS IN:
 a  -  CARTESIAN GRID
 b  -  SPHERE


 #  -  HOLE POINTS
 F  -  FRINGE POINTS (MODE 2 ONLY)
 ?  -  ORPHANED HOLE BOUNDARY POINTS
 +  -  ORPHANED OUTER BOUNDARY POINTS
 *  -  INTERPOLATION STENCILS UPDATING MORE THAN ONE MESH
 ?

MAP FOR CARTESIAN GRID

          PLANE  L = 13

     J         1         2         3         4         5         6         7
      12345678901234567890123456789012345678901234567890123456789012345678901
    K
    1 .........................................
    2 .........................................
    3 ...............b.b..b.b.b................
    4 ..........b.b..............b.b...........
    5 ...............BBBBBB....................
    6 ............BBB######B...................
    7 .......b...B#########B..........b........
    8 .....b....B##########BB...........b......
    9 ....b....B#############BB................
   10 ....b...B################B.........b.....
   11 ....b...B################B.........b.....
   12 ....b...B################B...............
   13 .....b...B#############BB.........b......
   14 .......b..B##########BB.........b........
   15 ...........B#########B...................
   16 ............BBB######B...................
   17 ............b..BBBBBB......b.............
   18 ...............b.b..b.b.b................
   19 .........................................
   20 .........................................
   21 .........................................
```

**Figure 16. Diagnostic map.**

```
INFORMATION FILE

                    LEGEND


    IBPNTS                - NUMBER OF BOUNDARY POINTS  IN A MESH
    IIPNTS                - NUMBER OF INTERPOLATION  STENCILS IN A MESH
    IISPTR, IIEPTR        - STARTING AND ENDING INDICES  IN QBC ARRAY OF
                            XMER3D
    I                     - BOUNDARY POINT NUMBER
    NM                    - RECIPIENT MESH
    JB, KB, LB            - BOUNDARY POINT INDICES
    ND                    - DONOR MESH
    JI, KI, LI            - STENCIL REFERENCE POINT
    DXINT, DYINT, DZINT   - INTERPOLATION COEFFICIENTS

    A  -  CARTESIAN GRID
    B  -  SPHERE


MESH NUMBER :     1  -  CARTESIAN GRID

    IBPNTS :   542
    IIPNTS :   441
    IISPTR :     1
    IIEPTR :   441


BOUNDARY POINT CONNECTIONS

    I  NM  JB  KB  LB    ND  JI  KI  LI     DXINT        DYINT        DZINT
    1  A   16   5  13    B   13   4  25   0.3941E+00   0.9755E+00   0.1387E+00
    2  A   17   5  13    B   12   4  24   0.9475E+00   0.9755E+00   0.2220E+00
    3  A   18   5  13    B   12   4  23   0.4808E+00   0.9755E+00   0.6366E+00
    4  A   19   5  13    B   11   4  23   0.9972E+00   0.9755E+00   0.5339E-01
    5  A   20   5  13    B   11   4  22   0.5017E+00   0.9755E+00   0.8566E+00
    6  A   21   5  13    B   11   4  22   0.1147E-07   0.9755E+00   0.6599E+00
    7  A   13   6  13    B   15   4  24   0.4213E-01   0.7872E+00   0.7375E+00
    8  A   14   6  13    B   14   4  23   0.6447E+00   0.7872E+00   0.3922E+00
    9  A   15   6  13    B   14   4  22   0.2138E+00   0.7872E+00   0.8190E-01
   10  A   22   6  13    B   10   4  18   0.4156E+00   0.7872E+00   0.3461E+00
   11  A   12   7  13    B   15   4  22   0.9804E+00   0.5240E+00   0.9256E+00
   12  A   22   7  13    B   10   4  13   0.3002E+00   0.5240E+00   0.8355E+00
   13  A   11   8  13    B   17   4  21   0.5923E-02   0.1297E+00   0.8832E+00
   14  A   22   8  13    B   10   4   9   0.1279E+00   0.1297E+00   0.3249E+00
   15  A   23   8  13    B    9   4   9   0.2871E+00   0.1297E+00   0.8527E+00
   16  A   10   9  13    B   17   3  22   0.9437E+00   0.5000E+00   0.2460E-01
   17  A   24   9  13    B    7   3   7   0.9280E+00   0.5000E+00   0.1771E+00
   18  A   25   9  13    B    7   3   8   0.1192E+00   0.5000E+00   0.4874E+00
   19  A    9  10  13    B   18   2  23   0.7027E+00   0.4760E+00   0.1472E+00
   20  A   26  10  13    B    5   2   7   0.6844E+00   0.4760E+00   0.9085E+00
   21  A    9  11  13    B   18   1  22   0.9515E+00   0.9197E-13   0.6408E+00
   22  A   26  11  13    B    5   1   7   0.2959E+00   0.9197E-13   0.4903E-01
```

**Figure 17. Information file.**

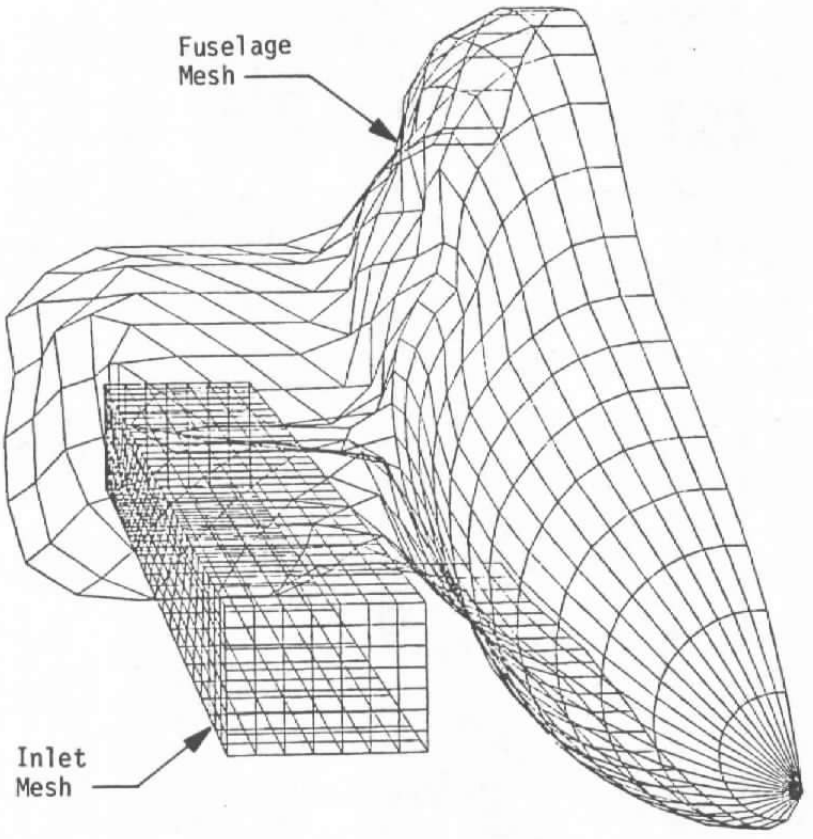


Figure 18. Inlet/forebody configuration.

```
C                         PEGSUS 3.0 INPUT

C                    FOREBODY/INLET CONFIGURATION
C                 ( BODY AND INLET MESHES INTERACTING)

C  OPTIONS
C   MODE=4 - DEFAULT(EVERYTHING)
 $OPTION MODE = 4,
             $END


C  DEFINE THE MESHES

 $MESH NAME = 'FUSE1',
C        FUSELAGE MESH OUTER BOUNDARY INTERPOLATES FROM INLET MESH
C  NOTE:  IN THIS CASE, A PEGSUS OUTER BOUNDARY IS DEFINED ON THE
C  SURFACE OF THE FUSELAGE]
       OBLINK1 = 'INLET',
                                  $END
 $MESH NAME = 'INLET',
C  INLET MESH OUTER BOUNDARY INTERPOLATES FROM FUSELAGE MESH
       OBLINK1 = 'FUSE1',
                                  $END


C  BOUNDARY DEFINITIONS

 $BOUNDARY  NAME = 'INLET OUTER BOUNDARY',
            ISPARTOF = 'INLET',
                                  $END
 $BOUNDARY  NAME = 'FUSE1 OUTER BOUNDARY',
            ISPARTOF = 'FUSE1',
                                  $END

C  SURFACE DEFINITIONS


C  FUSE1 MESH SURFACE DEFINITIONS
 $SURFACE ISPARTOF = 'FUSE1 OUTER BOUNDARY',
           JRANGE = 10,40,
           KRANGE = 2,42,
C        DEFINE THE FUSELAGE "OUTER BOUNDARY" ON THE SURFACE OF
C  THE FUSELAGE, AT THE INTERSECTION OF THE INLET AND FUSELAGE
C  MESHES
           LRANGE =  1, 1,
                                  $END


C  INLET OUTER SURFACE DEFINITIONS

 $SURFACE ISPARTOF = 'INLET OUTER BOUNDARY',
           JRANGE = 1,1,
```

**a. Page 1**

**Figure 19. Input for inlet/forebody.**

```
          KRANGE = 1,15,
          LRANGE = 1,15,
                                    $END
 $SURFACE ISPARTOF = 'INLET OUTER BOUNDARY',
          JRANGE = 1,20,
          KRANGE = 1,1,
          LRANGE = 1,15,
                                    $END
 $SURFACE ISPARTOF = 'INLET OUTER BOUNDARY',
          JRANGE = 1,20,
          KRANGE = 15,15,
          LRANGE = 1,15,
                                    $END
 $SURFACE ISPARTOF = 'INLET OUTER BOUNDARY',
          JRANGE = 1,20,
          KRANGE = 1,15,
          LRANGE = 1,1,
                                    $END
 $SURFACE ISPARTOF = 'INLET OUTER BOUNDARY',
          JRANGE = 1,20,
          KRANGE = 1,15,
          LRANGE = 15,15,
                                    $END

C  END OF FOREBODY/INLET PEGSUS 3.0 INPUT
```

**b. Page 2**
**Figure 19. Concluded.**

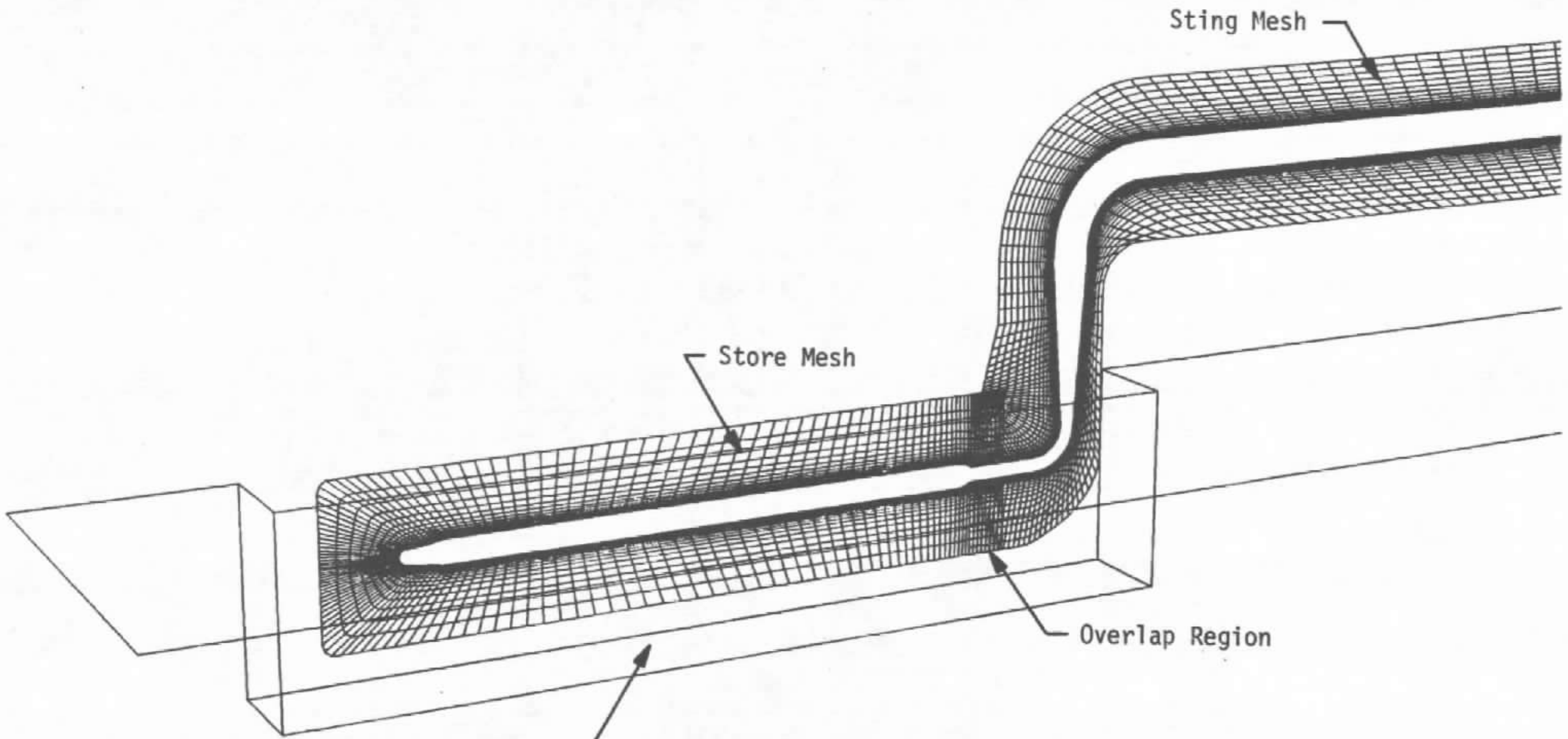


**Figure 20. Store/sting configuration.**

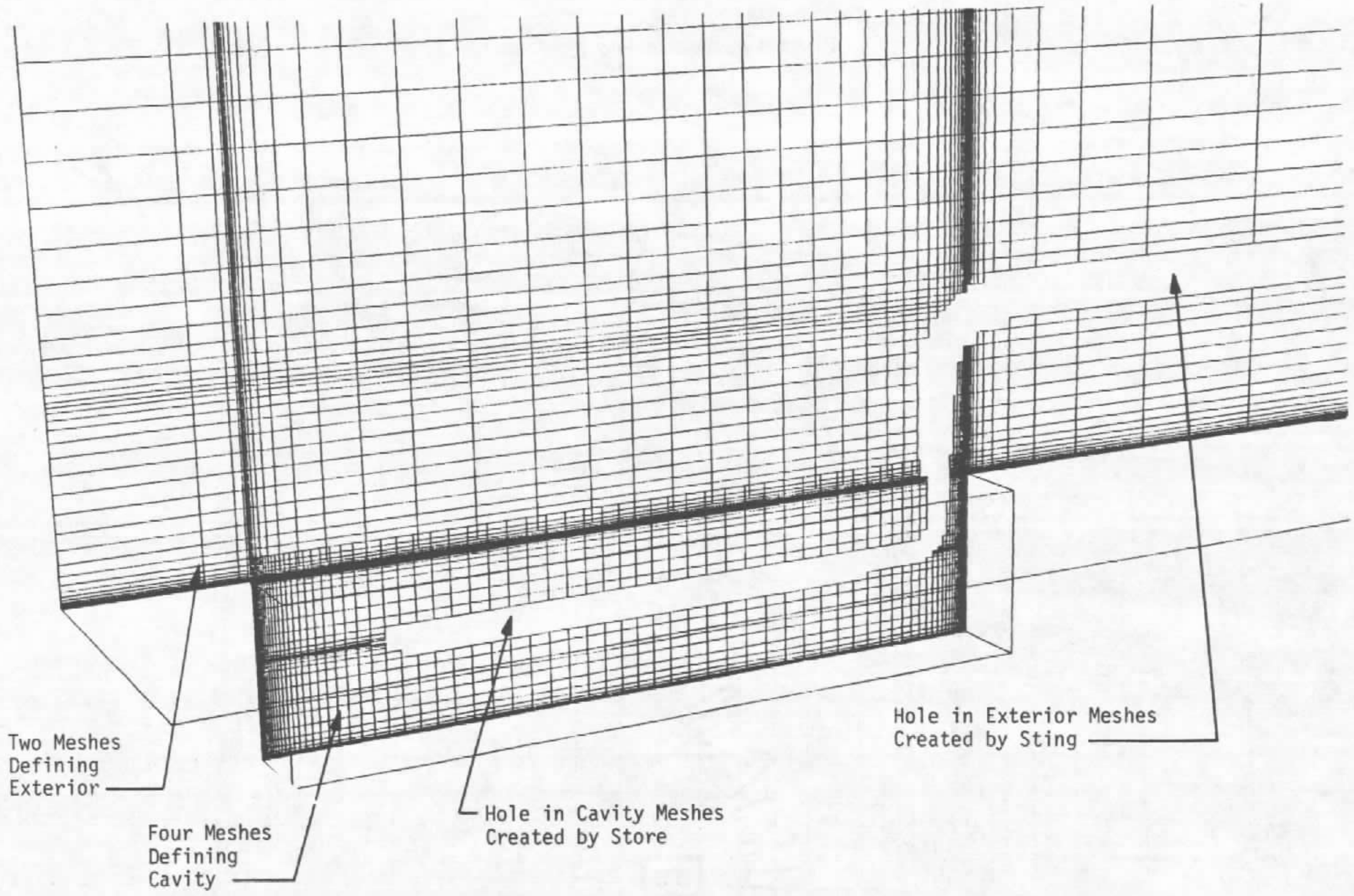

Two Meshes Defining Exterior
Four Meshes Defining Cavity
Hole in Cavity Meshes Created by Store
Hole in Exterior Meshes Created by Sting

**Figure 21. Holes in meshes created by store/sting.**

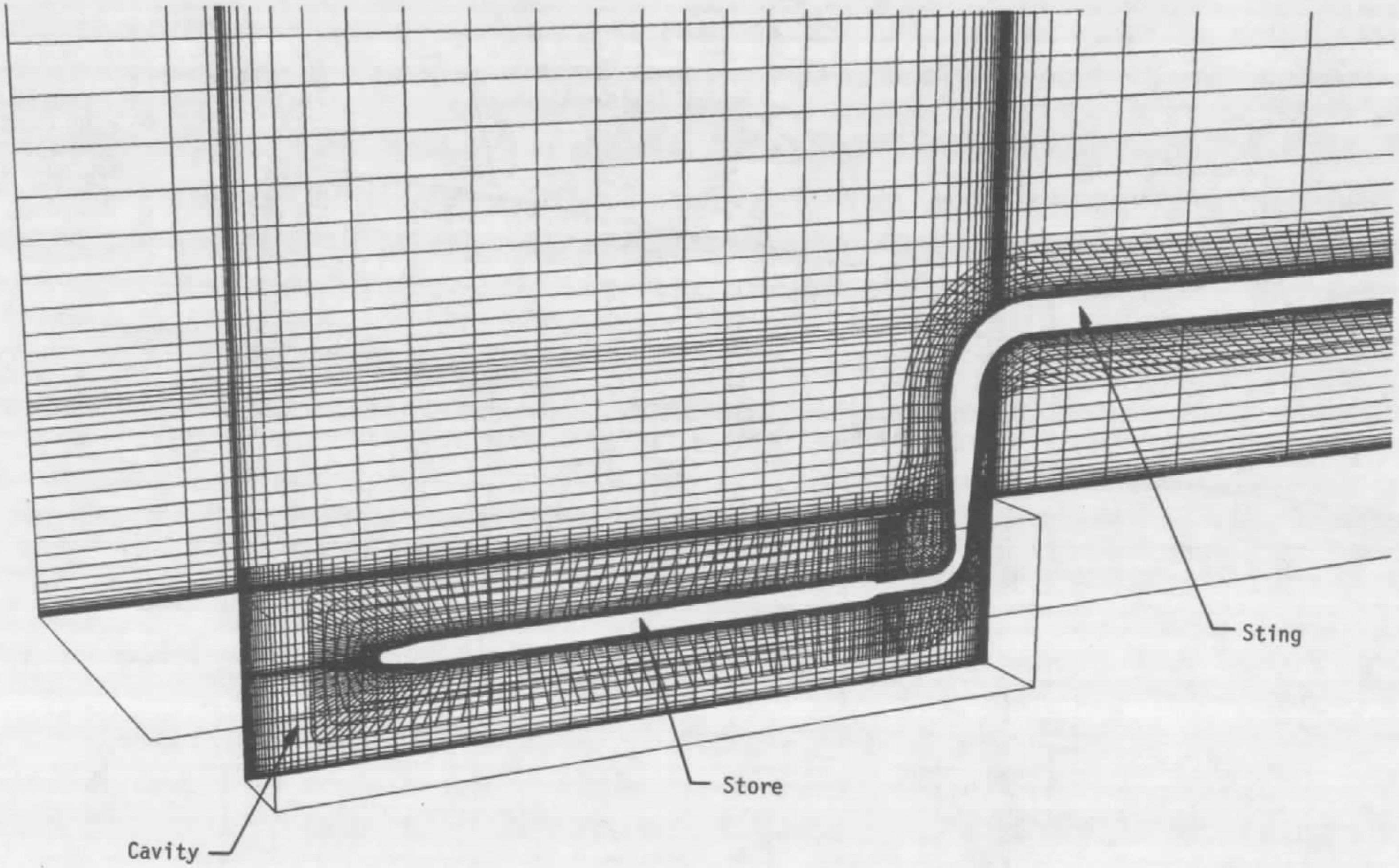


Figure 22. Store/sting in a cavity.

```
C                         PEGSUS 3.0
C                    STORE/STING IN CAVITY

C  FIGS. 20-22 MESHES:                            JMAX  KMAX  LMAX

C         STORE 9X                                  34    30    49
C         STING 9X                                  34    30    49

C         CAVITY A                                  26    30    65
C         CAVITY B                                  26    30    65
C         CAVITY C                                  26    30    65
C         CAVITY D                                  26    30    65

C         EXTERIOR A                                82    21    29
C         EXTERIOR B                                82    21    29


C  OPTIONS
 $OPTION
    MODE = 4,
              $END


C  MESH DEFINITIONS

 $MESH NAME = 'EXTERIOR A',
       OBLINK1 = 'CAVITY A',
       OBLINK2 = 'CAVITY B',
       OBLINK3 = 'CAVITY C',
       OBLINK4 = 'EXTERIOR B',
       HBLINK1 = 'CAVITY A',
       HBLINK2 = 'CAVITY B',
       HBLINK3 = 'CAVITY C',
       HBLINK4 = 'CAVITY D',
       HBLINK5 = 'STING 9X',
       XO = 11.0 ,YO = 0.0, ZO = 0.0,
                                $END
 $MESH NAME = 'EXTERIOR B',
       OBLINK1 = 'CAVITY B',
       OBLINK2 = 'CAVITY C',
       OBLINK3 = 'CAVITY D',
       OBLINK4 = 'EXTERIOR A',
       HBLINK1 = 'CAVITY A',
       HBLINK2 = 'CAVITY B',
       HBLINK3 = 'CAVITY C',
       HBLINK4 = 'CAVITY D',
       HBLINK5 = 'STING 9X',
       XO = 11.0 ,YO = 0.0, ZO = 0.0,
                                $END
 $MESH NAME = 'CAVITY A',
       OBLINK1 = 'CAVITY B',
       OBLINK2 = 'EXTERIOR A',
```

**a. Page 1**
**Figure 23. Input for store/sting in a cavity.**

```
      HBLINK1 = 'STORE 9X',
      XO = 11.0 ,YO = 0.0, ZO = 0.0,
                                   $END
 $MESH NAME = 'CAVITY B',
      OBLINK1 = 'CAVITY A',
      OBLINK2 = 'CAVITY C',
      OBLINK3 = 'EXTERIOR A',
      OBLINK4 = 'EXTERIOR B',
      HBLINK1 = 'STORE 9X',
      XO = 11.0 ,YO = 0.0, ZO = 0.0,
                                   $END
 $MESH NAME = 'CAVITY C',
      OBLINK1 = 'CAVITY B',
      OBLINK2 = 'CAVITY D',
      OBLINK3 = 'EXTERIOR A',
      OBLINK4 = 'EXTERIOR B',
      OBLINK5 = 'STING 9X',
      HBLINK1 = 'STORE 9X',
      HBLINK2 = 'STING 9X',
      XO = 11.0 ,YO = 0.0, ZO = 0.0,
                                   $END
 $MESH NAME = 'CAVITY D',
      OBLINK1 = 'CAVITY C',
      OBLINK2 = 'EXTERIOR B',
      OBLINK3 = 'STING 9X',
      HBLINK1 = 'STORE 9X',
      HBLINK2 = 'STING 9X',
      XO = 11.0 ,YO = 0.0, ZO = 0.0,
                                   $END
 $MESH NAME = 'STORE 9X',
      OBLINK1 = 'STING 9X',
      OBLINK2 = 'CAVITY A',
      OBLINK3 = 'CAVITY B',
      OBLINK4 = 'CAVITY C',
      OBLINK5 = 'CAVITY D',
      XO = 17.9316 ,YO = 0.0, ZO = -1.8,
                                   $END
 $MESH NAME = 'STING 9X',
      OBLINK1 = 'STORE 9X',
      OBLINK2 = 'CAVITY C',
      OBLINK3 = 'CAVITY D',
      OBLINK4 = 'EXTERIOR B',
      OBLINK5 = 'EXTERIOR A',
      XO = 17.9316 ,YO = 0.0, ZO = -1.8,
                                   $END

C  BOUNDARY DEFINITIONS

 $BOUNDARY  NAME = 'CAVITY A HOLE BOUNDARY',
            MHOLEIN1 = 'EXTERIOR A',
            MHOLEIN2 = 'EXTERIOR B',
            ISPARTOF = 'CAVITY A',
                                   $END
```

**b. Page 2**
**Figure 23. Continued.**

```
$BOUNDARY  NAME = 'CAVITY B HOLE BOUNDARY',
           MHOLEIN1 = 'EXTERIOR A',
           MHOLEIN2 = 'EXTERIOR B',
           ISPARTOF = 'CAVITY B',
                                     $END

$BOUNDARY  NAME = 'CAVITY C HOLE BOUNDARY',
           MHOLEIN1 = 'EXTERIOR A',
           MHOLEIN2 = 'EXTERIOR B',
           ISPARTOF = 'CAVITY C',
                                     $END

$BOUNDARY  NAME = 'CAVITY D HOLE BOUNDARY',
           MHOLEIN1 = 'EXTERIOR A',
           MHOLEIN2 = 'EXTERIOR B',
           ISPARTOF = 'CAVITY D',
                                     $END

$BOUNDARY  NAME = 'STORE 9X HOLE BOUNDARY',
           MHOLEIN1 = 'CAVITY A',
           MHOLEIN2 = 'CAVITY B',
           MHOLEIN3 = 'CAVITY C',
           MHOLEIN4 = 'CAVITY D',
           ISPARTOF = 'STORE 9X',
                                     $END

$BOUNDARY  NAME = 'STING 9X HOLE BOUNDARY',
           MHOLEIN1 = 'CAVITY C',
           MHOLEIN2 = 'CAVITY D',
           MHOLEIN3 = 'EXTERIOR B',
           MHOLEIN4 = 'EXTERIOR A',
           ISPARTOF = 'STING 9X',
                                     $END

$BOUNDARY  NAME = 'EXTERIOR A OUTER BOUNDARY',
           ISPARTOF = 'EXTERIOR A',
                                     $END

$BOUNDARY  NAME = 'EXTERIOR B OUTER BOUNDARY',
           ISPARTOF = 'EXTERIOR B',
                                     $END

$BOUNDARY  NAME = 'CAVITY A OUTER BOUNDARY',
           ISPARTOF = 'CAVITY A',
                                     $END

$BOUNDARY  NAME = 'CAVITY B OUTER BOUNDARY',
           ISPARTOF = 'CAVITY B',
                                     $END

$BOUNDARY  NAME = 'CAVITY C OUTER BOUNDARY',
           ISPARTOF = 'CAVITY C',
                                     $END
```

**c. Page 3**
**Figure 23. Continued.**

```
 $BOUNDARY  NAME = 'CAVITY D OUTER BOUNDARY',
            ISPARTOF = 'CAVITY D',
                                        $END

 $BOUNDARY  NAME = 'STORE 9X OUTER BOUNDARY',
            ISPARTOF = 'STORE 9X',
                                        $END

 $BOUNDARY  NAME = 'STING 9X OUTER BOUNDARY',
            ISPARTOF = 'STING 9X',
                                        $END


C  SURFACE DEFINITIONS

 $SURFACE ISPARTOF = 'CAVITY A HOLE BOUNDARY',
      JRANGE = 5,5, KRANGE = 5,30, LRANGE = 44,61, NVOUT = '-J',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY A HOLE BOUNDARY',
      JRANGE = 26,26, KRANGE = 5,30, LRANGE = 44,61, NVOUT = '+J',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY A HOLE BOUNDARY',
      JRANGE = 5,26, KRANGE = 5,5, LRANGE = 44,61, NVOUT = '-K',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY A HOLE BOUNDARY',
      JRANGE = 5,26, KRANGE = 5,30, LRANGE = 61,61, NVOUT = '+L',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY B HOLE BOUNDARY',
      JRANGE = 1,1, KRANGE = 5,30, LRANGE = 44,61, NVOUT = '-J',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY B HOLE BOUNDARY',
      JRANGE = 26,26, KRANGE = 5,30, LRANGE = 44,61, NVOUT = '+J',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY B HOLE BOUNDARY',
      JRANGE = 1,26, KRANGE = 5,5, LRANGE = 44,61, NVOUT = '-K',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY B HOLE BOUNDARY',
      JRANGE = 1,26, KRANGE = 5,30, LRANGE = 61,61, NVOUT = '+L',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY C HOLE BOUNDARY',
      JRANGE = 1,1, KRANGE = 5,30, LRANGE = 44,61, NVOUT = '-J',
                                        $END

 $SURFACE ISPARTOF = 'CAVITY C HOLE BOUNDARY',
      JRANGE = 26,26, KRANGE = 5,30, LRANGE = 44,61, NVOUT = '+J',
                                        $END
```

**d. Page 4**
**Figure 23. Continued.**

```
$SURFACE ISPARTOF = 'CAVITY C HOLE BOUNDARY',
    JRANGE = 1,26, KRANGE = 5,5, LRANGE = 44,61, NVOUT = '-K',
                            $END

$SURFACE ISPARTOF = 'CAVITY C HOLE BOUNDARY',
    JRANGE = 1,26, KRANGE = 5,30, LRANGE = 61,61, NVOUT = '+L',
                            $END

$SURFACE ISPARTOF = 'CAVITY D HOLE BOUNDARY',
    JRANGE = 1,1, KRANGE = 5,30, LRANGE = 44,61, NVOUT = '-J',
                            $END

$SURFACE ISPARTOF = 'CAVITY D HOLE BOUNDARY',
    JRANGE = 22,22, KRANGE = 5,30, LRANGE = 44,61, NVOUT = '+J',
                            $END

$SURFACE ISPARTOF = 'CAVITY D HOLE BOUNDARY',
    JRANGE = 1,22, KRANGE = 5,5, LRANGE = 44,61, NVOUT = '-K',
                            $END

$SURFACE ISPARTOF = 'CAVITY D HOLE BOUNDARY',
    JRANGE = 1,22, KRANGE = 5,30, LRANGE = 61,61, NVOUT = '+L',
                            $END

$SURFACE ISPARTOF = 'STORE 9X HOLE BOUNDARY',
    JRANGE = 1,82, KRANGE = 2,20, LRANGE = 15,15, NVOUT = '+L',
                            $END

$SURFACE ISPARTOF = 'STORE 9X HOLE BOUNDARY',
    JRANGE = 82,82, KRANGE = 2,20, LRANGE = 1,15, NVOUT = '+J',
                            $END

$SURFACE ISPARTOF = 'STING 9X HOLE BOUNDARY',
    JRANGE = 1,82, KRANGE = 2,20, LRANGE = 15,15, NVOUT = '+L',
                            $END

$SURFACE ISPARTOF = 'STING 9X HOLE BOUNDARY',
    JRANGE = 1,1, KRANGE = 2,20, LRANGE = 1,15, NVOUT = '-J',
                            $END

$SURFACE ISPARTOF = 'STING 9X HOLE BOUNDARY',
    JRANGE = 82,82, KRANGE = 2,20, LRANGE = 1,15, NVOUT = '+J',
                            $END


$SURFACE ISPARTOF = 'EXTERIOR A OUTER BOUNDARY',
    JRANGE = 34,34, KRANGE = 1,29, LRANGE = 2,48,
                            $END

$SURFACE ISPARTOF = 'EXTERIOR B OUTER BOUNDARY',
    JRANGE = 1,1, KRANGE = 1,29, LRANGE = 2,48,
                            $END
```

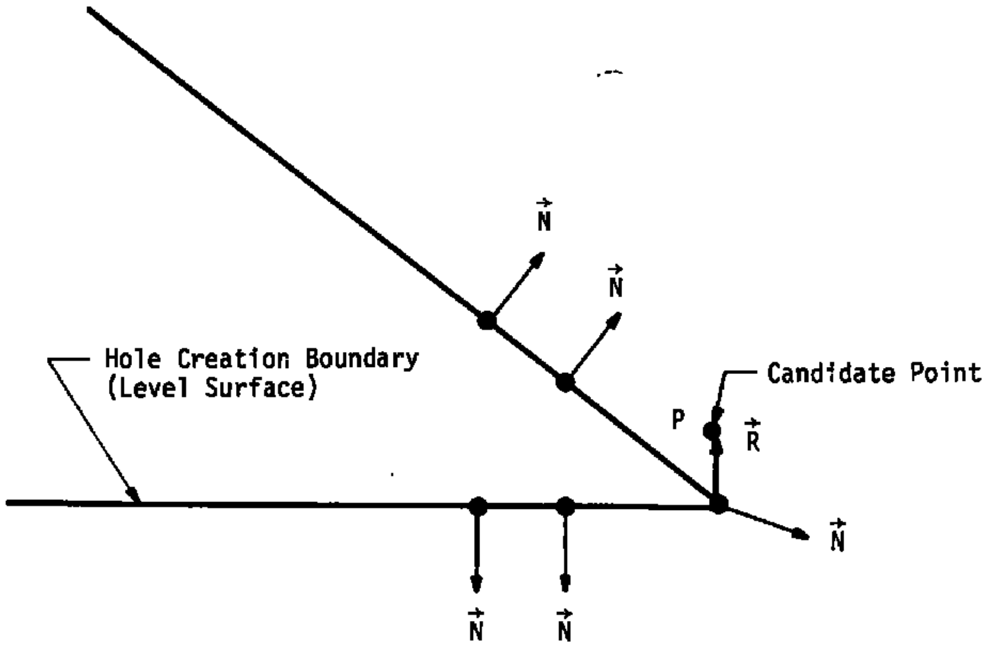
**e. Page 5**
**Figure 23. Continued.**

```
$SURFACE ISPARTOF = 'CAVITY A OUTER BOUNDARY',
    JRANGE = 1,1, KRANGE = 1,29, LRANGE = 45,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY A OUTER BOUNDARY',
    JRANGE = 26,26, KRANGE = 1,29, LRANGE = 1,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY A OUTER BOUNDARY',
    JRANGE = 2,25, KRANGE = 1,1, LRANGE = 45,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY A OUTER BOUNDARY',
    JRANGE = 2,25, KRANGE = 2,29, LRANGE = 65,65,
                                   $END


$SURFACE ISPARTOF = 'CAVITY B OUTER BOUNDARY',
    JRANGE = 1,1, KRANGE = 1,29, LRANGE = 1,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY B OUTER BOUNDARY',
    JRANGE = 26,26, KRANGE = 1,29, LRANGE = 1,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY B OUTER BOUNDARY',
    JRANGE = 2,25, KRANGE = 1,1, LRANGE = 45,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY B OUTER BOUNDARY',
    JRANGE = 2,25, KRANGE = 2,29, LRANGE = 65,65,
                                   $END


$SURFACE ISPARTOF = 'CAVITY C OUTER BOUNDARY',
    JRANGE = 1,1, KRANGE = 1,29, LRANGE = 1,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY C OUTER BOUNDARY',
    JRANGE = 26,26, KRANGE = 1,29, LRANGE = 1,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY C OUTER BOUNDARY',
    JRANGE = 2,25, KRANGE = 1,1, LRANGE = 45,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY C OUTER BOUNDARY',
    JRANGE = 2,25, KRANGE = 2,29, LRANGE = 65,65,
                                   $END


$SURFACE ISPARTOF = 'CAVITY D OUTER BOUNDARY',
    JRANGE = 1,1, KRANGE = 1,29, LRANGE = 1,65,
                                   $END
```

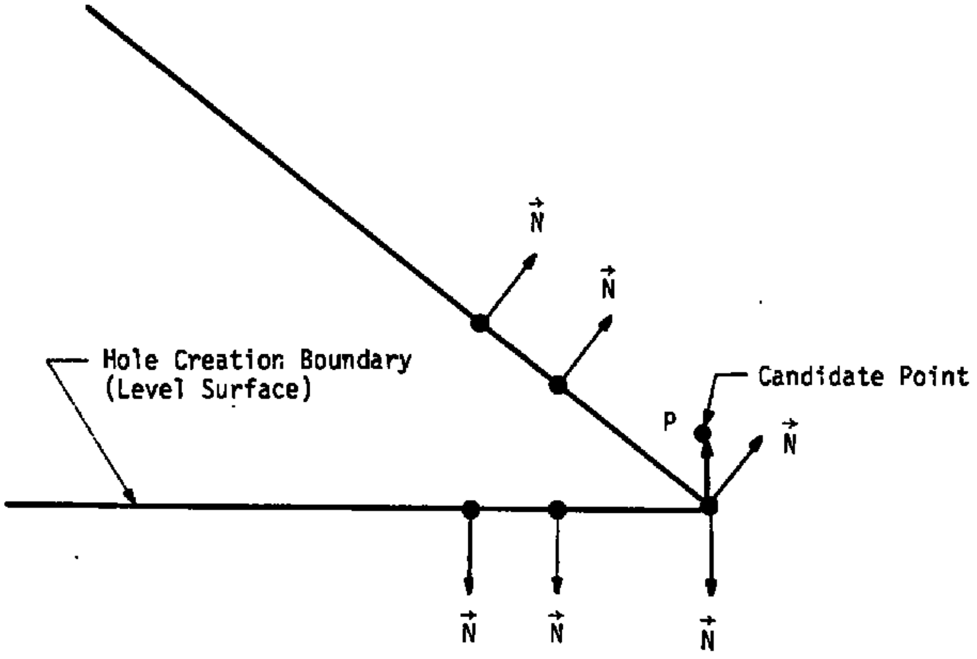
**f. Page 6**
**Figure 23. Continued.**

```
$SURFACE ISPARTOF = 'CAVITY D OUTER BOUNDARY',
    JRANGE = 26,26, KRANGE = 1,29, LRANGE = 45,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY D OUTER BOUNDARY',
    JRANGE = 2,25, KRANGE = 1,1, LRANGE = 45,65,
                                   $END

$SURFACE ISPARTOF = 'CAVITY D OUTER BOUNDARY',
    JRANGE = 2,25, KRANGE = 2,29, LRANGE = 65,65,
                                   $END


$SURFACE ISPARTOF = 'STORE 9X OUTER BOUNDARY',
    JRANGE = 1,82, KRANGE = 2,20, LRANGE = 29,29,
                                   $END

$SURFACE ISPARTOF = 'STORE 9X OUTER BOUNDARY',
    JRANGE = 82,82, KRANGE = 2,20, LRANGE = 2,28,
                                   $END

$SURFACE ISPARTOF = 'STING 9X OUTER BOUNDARY',
    JRANGE = 1,82, KRANGE = 2,20, LRANGE = 29,29,
                                   $END

$SURFACE ISPARTOF = 'STING 9X OUTER BOUNDARY',
    JRANGE = 1,1, KRANGE = 2,20, LRANGE = 2,28,
                                   $END

C  END OF STORE/STING IN A CAVITY PEGSUS 3.0 INPUT
```

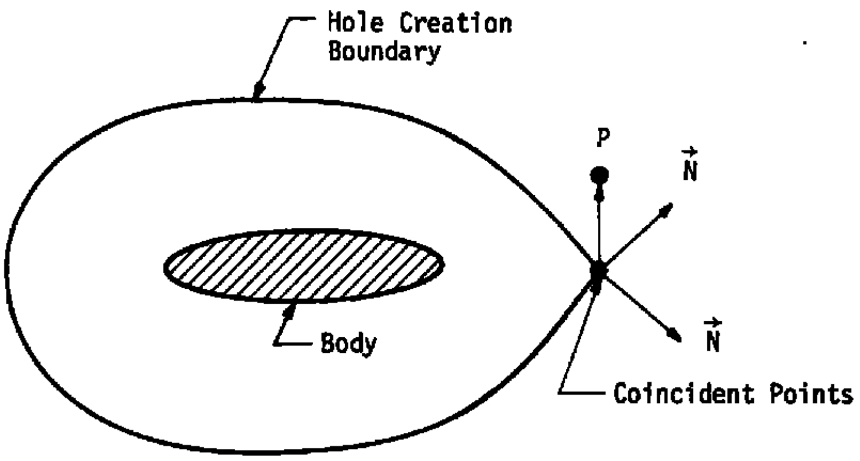
**g. Page 7**
**Figure 23. Concluded.**



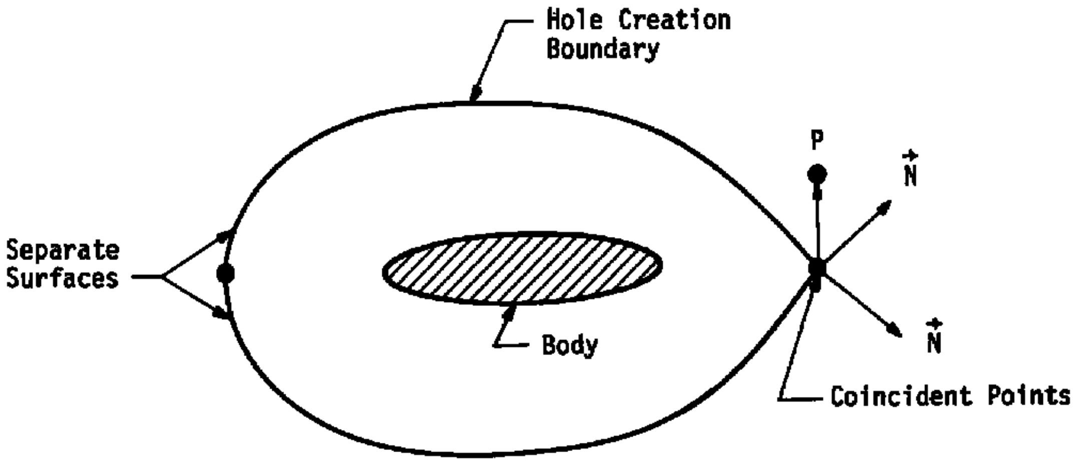
a. Problem: spurious hole points near sharp corner



b. Solution: divide hole creation boundary into two surfaces
Figure 24. Hole creation boundary with sharp corner.

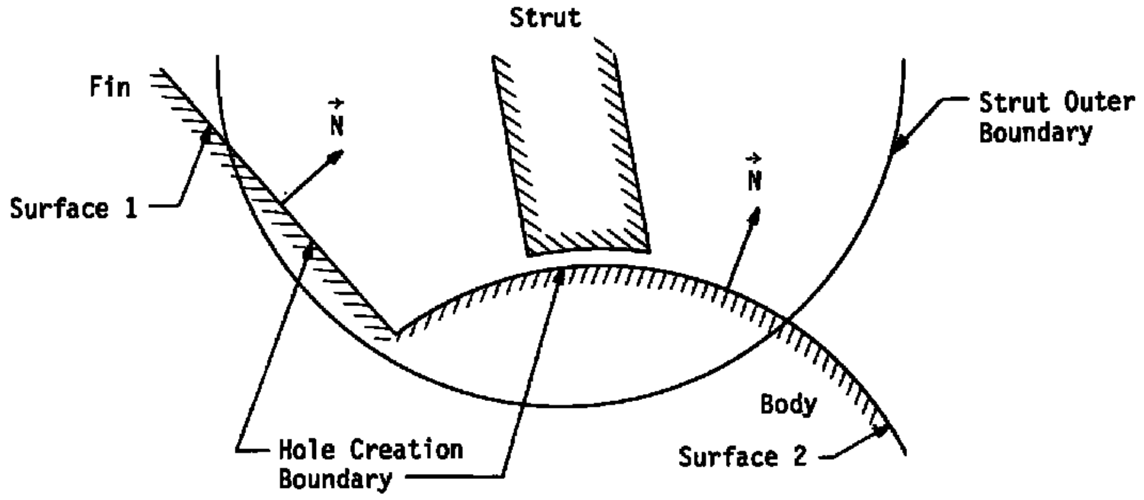


a. Problem: spurious hole points near coincident points

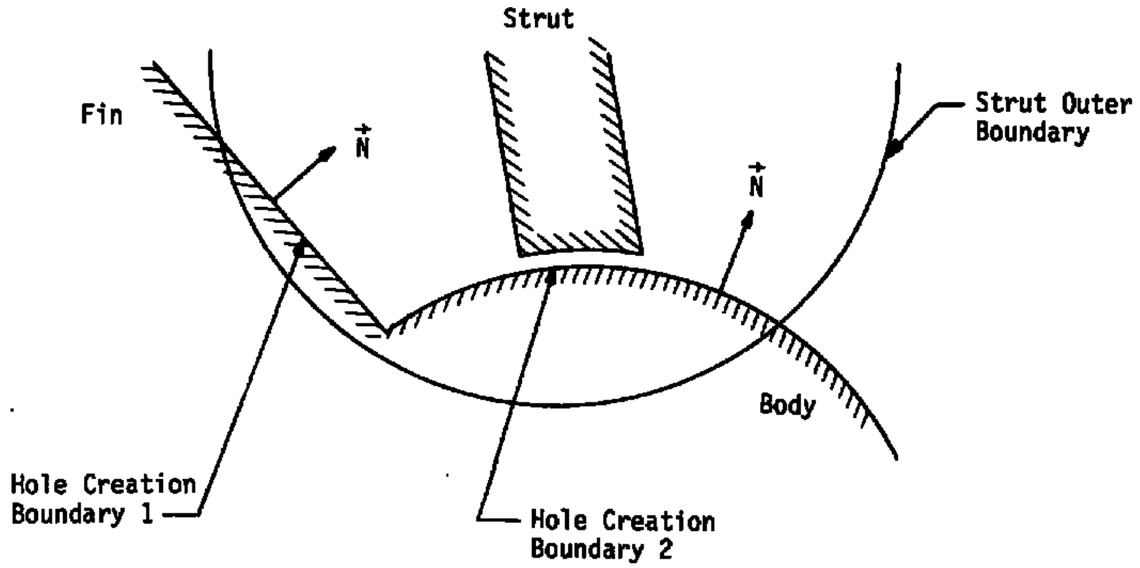


b. Solution: divide hole creation boundary into two surfaces

Figure 25. Single boundary with coincident points.

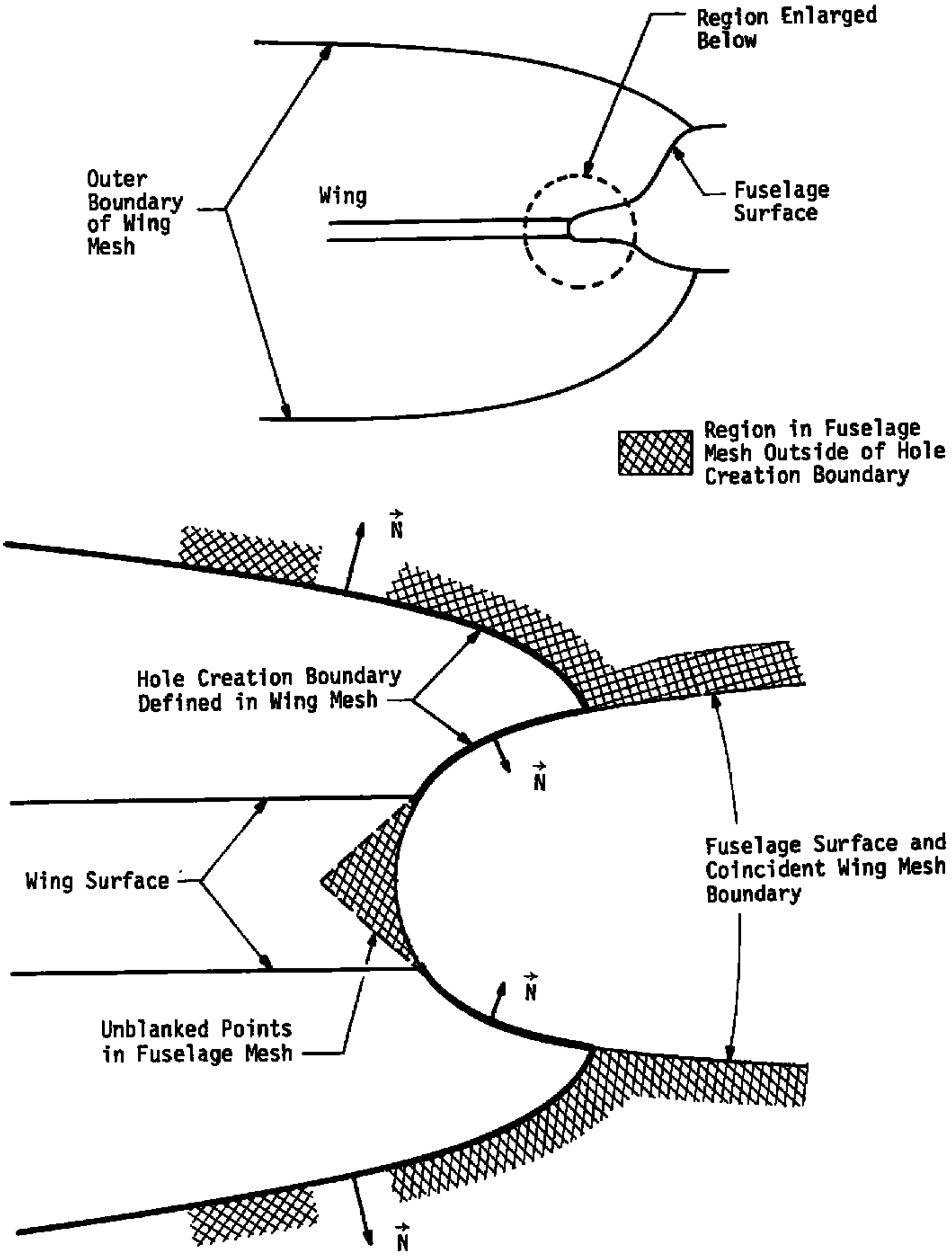


**a. Problem: improper hole blanking**

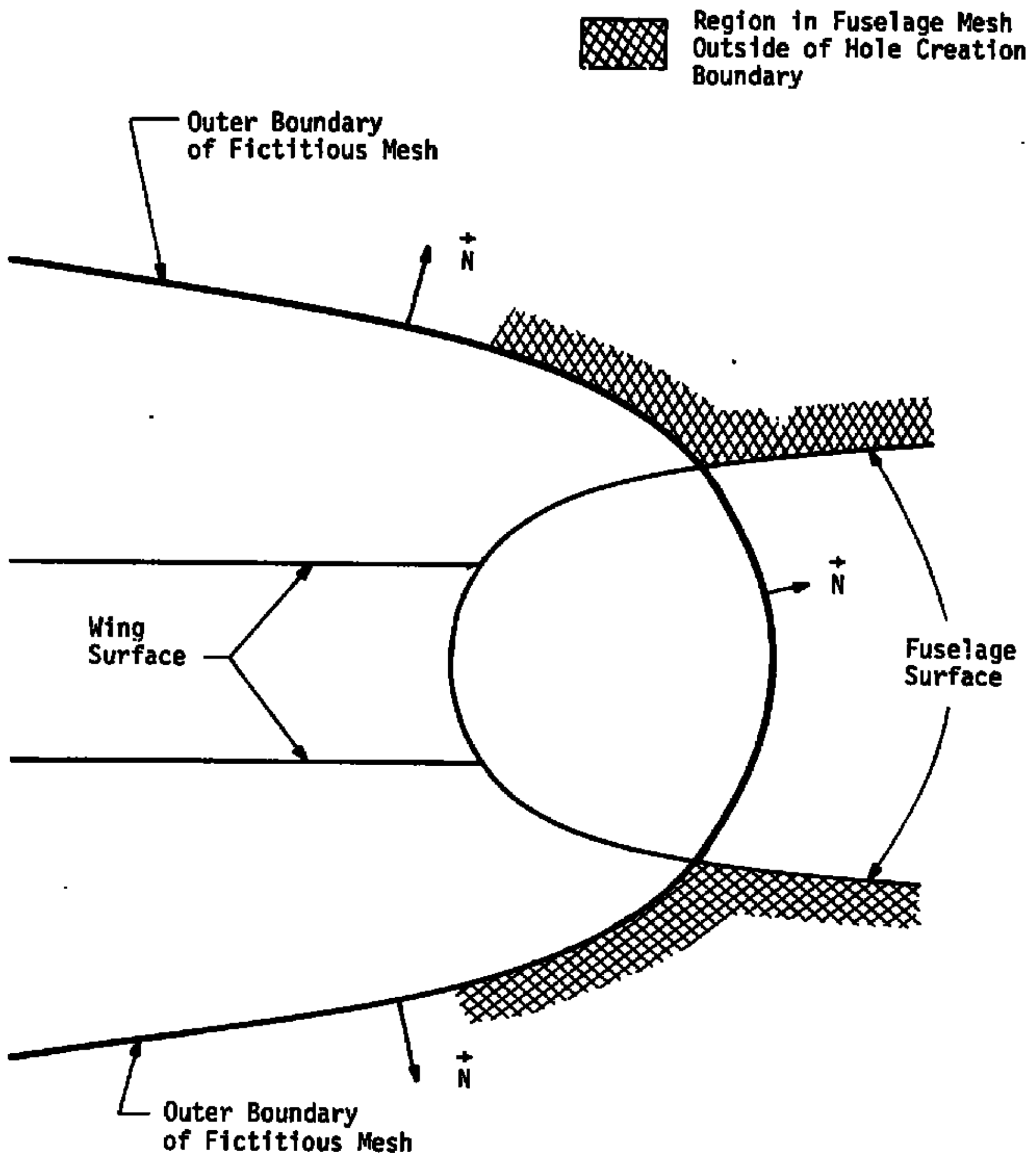


**b. Solution: divide hole creation boundary into two boundaries**

**Figure 26. Concave hole creation boundary.**


Region Enlarged Below
Outer Boundary of Wing Mesh
Wing
Fuselage Surface
▨ Region in Fuselage Mesh Outside of Hole Creation Boundary
$\vec{N}$
Hole Creation Boundary Defined in Wing Mesh
$\vec{N}$
Wing Surface
Fuselage Surface and Coincident Wing Mesh Boundary
$\vec{N}$
Unblanked Points in Fuselage Mesh
$\vec{N}$

**a. Problem: unblanked hole points near wing/body juncture**
**Figure 27. Complex wing/body juncture.**


▨ Region in Fuselage Mesh Outside of Hole Creation Boundary
Outer Boundary of Fictitious Mesh
$\vec{N}$
Wing Surface
$\vec{N}$
Fuselage Surface
$\vec{N}$
Outer Boundary of Fictitious Mesh

**b. Solution: use of fictitious mesh to create holes**
**Figure 27. Concluded.**

## APPENDIX A
## GLOSSARY OF TERMS

| | |
|---|---|
| Blanked point | Mesh point excluded from the flow solution. Hole points, interpolated boundary points, and orphaned hole points are blanked. |
| Boundary | Collection of one or more surfaces. |
| Candidate point | Boundary point for which an interpolation element in another mesh is to be found. |
| Coincident points | Points in different meshes which are closer than a predefined tolerance. |
| Composite mesh | Single file containing all meshes in a multiple-mesh configuration. |
| Donor mesh | Mesh that sends information by interpolation to another mesh. |
| Fringe point | Point on a hole boundary (synonymous with hole boundary point). |
| Hole | Collection of hole points, i.e. points in a mesh that have been blanked. |
| Hole boundary | Collection of fringe points surrounding a hole. |
| Hole boundary point | Point on a hole boundary, by definition adjacent to a hole point. |
| Hole creation boundary | Collection of surfaces in one mesh that creates a hole in another mesh. |
| Hole point | Mesh point that has been blanked out. |
| Interpolation file | File consisting of (1) boundary points, (2) pointers to interpolation stencils, and (3) interpolation coefficients for each mesh. The interpolation file is output by PEGSUS and is input by the XMER3D flow solver. |

| | |
|---|---|
| Orphan point | Boundary point that cannot be legally interpolated from any other mesh. Orphaned hole boundary points are always undesirable; orphaned outer boundary points may or may not be important (an outer boundary of a mesh may be only partially embedded in another mesh). |
| Recipient mesh | Mesh that receives information from another mesh. |
| Stencil | Point and its immediate neighbors that defines a fringe point; e.g., if any neighbor is a hole point, then the central point is a fringe point. Earlier versions of PEGSUS used a 27-point stencil; PEGSUS 3.0 uses a 7-point stencil. Also, an interpolation element is often referred to as a stencil in chimera terminology. |
| Surface | Part of a level surface of a mesh. Collections of one or more surfaces comprise a boundary. |

# APPENDIX B
# GLOBAL VARIABLES

The following data structures are generated by PEGSUS 3.0.

## COMMON/BOUNPARM/

| | |
|---|---|
| BNAME | Dimensioned [5,NBDIM]. BNAME(1..5,N) contains the alphanumeric name of Boundary N. |
| BPARTOF | Dimensioned [NBDIM]. BPARTOF(N) is the mesh number of the mesh that contains Boundary N. |
| BTYPE | Dimensioned [NBDIM]. BTYPE(N) is the type of Boundary N. The type is a character string, either 'HOLE' or 'OUTER'. |
| MHOLEIN | Dimensioned [NBDIM,30]. MHOLEIN(N,M) is the mesh number of the M$^{th}$ mesh in which hole creation Boundary N causes a hole. |
| NBOUN | Total number of boundaries in composite mesh. |
| NHOLEIN | Dimensioned [NBDIM]. NHOLEIN(N) is the total number of meshes in which hole creation Boundary N makes a hole. |

## COMMON/INTERP/

| | |
|---|---|
| DXINT, DYINT, DZINT | Dimensioned [MSLEN]. DXINT, DYINT, and DZINT are interpolation coefficients for a given mesh. |
| JI, KI, LI | Dimensioned [MSLEN]. JI, KI, and LI are indices of interpolation stencils for a given mesh. |
| JB, KB, LB | Dimensioned [MSLEN]. JB, KB, and LB are indices of boundary points for a given mesh. |
| IBC | Dimensioned [MSLEN]. Pointer from boundary points to associated interpolation stencils. |
| IBLANK | Dimensioned [MLEN]. Mesh IBLANK array. |

| | |
|---|---|
| IIPNTS | Dimensioned [MDIM]. IIPNTS(M) is the number of interpolation stencils in Mesh M. |
| IBPNTS | Dimensioned [MDIM]. IBPNTS(M) is the number of boundary points in Mesh M. |

COMMON/LINKS/

| | |
|---|---|
| MHBLINK | Dimensioned [MDIM,30]. MHBLINK(M,N) is the mesh number coresponding to the Nth hole boundary link of Mesh M. |
| MOBLINK | DImensioned [MDIM,30]. MOBLINK(M,N) is the mesh number corresponding to the Nth outer boundary link of Mesh M. |
| NHBLINK | Dimensioned [MDIM]. NHBLINK(M) is the number of hole boundary links associated with Mesh M. |
| NOBLINK | Dimensioned [MDIM]. NOBLINK(M) is the number of outer boundary links associated with Mesh M. |

COMMON/MESH/

| | |
|---|---|
| X,Y,Z | Dimensioned [MLEN]. Mesh coordinate array. |

COMMON/MESHLINK/

| | |
|---|---|
| IHOLE | Dimensioned [MDIM]. IHOLE(N) is the ID number of the first hole creation boundary defined in Mesh N. |
| IOUTER | Dimensioned [MDIM]. IOUTER(N) is the ID number of the outer boundary of Mesh N. |

COMMON/MESHPARM/

| | |
|---|---|
| NMESH | Total number of meshes in composite mesh. |
| NAME | Dimensioned [5,MDIM]. NAME(1..5,N) contains the name of the Nth mesh as an alphanumeric string. |
| MJMAX, | Dimensioned [MDIM]. Mesh dimensions. MJMAX(N) is the |

| | |
|---|---|
| MKMAX, MLMAX | maximum J dimension of Mesh N. |
| NSREGH | Dimensioned [MDIM]. NSREGH(N) is the number of hole search regions in Mesh N. |
| MROFF | Dimensioned [MDIM]. Mesh offset. MROFF(N) is the absolute index of the first point in the Nth mesh in the composite mesh. |

**COMMON/OPTION/**

| | |
|---|---|
| MODE | Integer variable equal to 1 through 7, corresponding to the mode in which PEGSUS is being run. |

**COMMON/PLOT/**

| | |
|---|---|
| IBPLOT | Dimensioned [MLEN]. IBPLOT is an IBLANK array in which blanked hole boundary points are identified with the mesh that creates the hole. IBPLOT is used with PLOT3D. |

**COMMON/SURF/**

| | |
|---|---|
| XS,YS,ZS | Dimensioned [MSLEN]. Surface coordinate array. |
| VNX,VNY,VNZ | Dimensioned [MSLEN]. Array of normal vector components. |
| ISURF | Dimensioned [MSLEN]. Absolute index of point in mesh at which a normal vector has been calculated. |

**COMMON/SURFPARM/**

| | |
|---|---|
| NSURF | Total number of surfaces in composite mesh. |
| JRANGE, KRANGE, LRANGE | Integer ranges of surfaces. Dimensioned [2,NSDIM]. |

JRANGE(1,N) = 5
JRANGE(2,N) = 15
KRANGE(1,N) = 10
KRANGE(2,N) = 30

LRANGE(1,N) = 5
LRANGE(2,N) = 5

defines Surface N to be a level surface at L = 5 ranging between J = 5 and 15, and K = 10 and 30.

| | |
|---|---|
| NVOUT | Dimensioned [NSDIM]. Denotes the outward direction in coordinate space of a surface. Must be '−J', '+J', '−K', '+K', '−L', or '+L'. NVOUT(N) is the outward direction of Surface N. |
| ISPARTOF | Dimensioned [5,NSDIM]. ISPARTOF(1..5,N) is the alphanumeric name of the boundary of which Surface N is a member. |

**COMMON/SURFLINK/**

| | |
|---|---|
| IDSURF | Dimensioned [NBDIM,NSDIM]. IDSURF(N,M) is the ID number of the M^th surface which comprises Boundary N. |
| NUSURF | Dimensioned [NBDIM]. NUSURF(N) is the number of surfaces comprising Boundary N. |

**COMMON/TOLS/**

| | |
|---|---|
| DELTA | Dimensioned [MDIM]. Points in the composite mesh within a distance DELTA(M) of any point in Mesh M will be considered coincident. |
| XMAX<br>YMAX<br>ZMAX | Dimensioned [MDIM]. Maximum X,Y,Z values of each mesh. |
| XMIN<br>YMIN<br>ZMIN | Dimensioned [MDIM]. Minimum X,Y,Z values of each mesh. |

**COMMON/UTIL/**

| | |
|---|---|
| ALPHA<br>BETA<br>GAMA | Dimensioned [MDIM]. Euler angles of each mesh. |

| | |
|---|---|
| XR<br>YR<br>ZR | Dimensioned [MDIM]. Point about which mesh is to be rotated. |
| X0<br>Y0<br>Z0 | Dimensioned [MDIM]. Amount by which mesh is to be translated. |
| SCALE | Dimensioned [MDIM]. Scaling factor for each mesh. |

# APPENDIX C
# PROGRAM STRUCTURE

The following chart depicts the hierarchical structure of PEGSUS 3.0. Modules with names that start in the same column are called from the same level; e.g., the top-level module PEGSUS30 directly calls PREPROC, INPUT, COMPOS, INITDAF, CRIBLANK, PRCHOLES, PRCOUTER, XMSPECS, WRITEOUT, and DIAGOUT. Modules such as NUMSPTS obviously are primitive functions or subroutines that do not call other modules.

```
PEGSUS30
  PREPROC
      STRIP
      SYNCHK
      PARAMCHK
  INPUT
      MESHINP
          GETMESH
              MESHDEF
              MESHNO
                  CONVERT
                  EQUAL
              CONVERT
          MESHCHK
          READMESH
              EQUAL
              MESHCHK2
                  MESHNO
              SCALM
              TRANS
              ROTATE
              FDELTA
          MESHDEF
          MESHSTAT
      BOUNINP
          MESHNO
          CONVERT
          BOUNDEF
      GETSURF
          CONVERT
          SURFDEF
```

```
            BOUNDNO
                  CONVERT
                  EQUAL
      DWNPTRS
      SURFGEN
            CONVERT
            JKLTOI
            NORMAL
                  JKLTOI
COMPOS
INITDAF
CRIBLANK
      HBIBLANK
            CONVERT
            JKLTOI
            DELLST
                  BGATHER
                  ISINSIDE
                  NUMSPTS
            NWIBLANK
            NWIDIAG
            OPTSRCH
                  ITOJKL
      FFRNGPTS
            CONVERT
            DELLST
            FNDNBRS
                  JKLTOI
                  ITOJKL
      OBILANK
            CONVERT
            NUMSPTS
PRCHOLES
      CONVERT
      DELLST
      ITOJKL
      INTPTS
            BGATHER
      INBOX
      TRILIN
```

```
            SRPINIT2
            INTDAT
                NEWTON
            TSTCOPTS
                CONVERT
                JKLTOI
            TSTCRNRS
                CONVERT
                JKLTOI
        DIAGIN
            JKLTOI
    PRCOUTER
        CONVERT
        DELLST
        ITOJKL
        INTPTS
        INBOX
        TRILIN
        DIAGIN
    XMSPECS
        NWIBLANK
    WRITEOUT
        WRTINFO
            CONVERT
            JKLTOI
    DIAGOUT
        CONVERT
        JKLTOI
```

## APPENDIX D
## SUBROUTINE DESCRIPTIONS

The following are descriptions of the program modules depicted in the hierarchical chart in Appendix C.

| | |
|---|---|
| BGATHER | Performs modified GATHER function. Unlike CRAY® version, target and source arrays may have different dimensions. BGATHER is also faster than the CRAY® GATHER function. |
| BOUNDEF | Defines default values for variables in $BOUNDARY NAMELIST records. |
| BOUNDNO | Given a boundary name; returns the boundary number. |
| BOUNINP | Inputs boundary information from NAMELIST and checks for consistency. |
| COMPOS | Writes out composite mesh. |
| CONVERT | Extracts a mesh or boundary name from an array of names. |
| CRIBLANK | Blanks out hole and outer boundary points in each mesh. |
| DELLST | Utility subroutine. Deletes all occurrences of a number from a list, and returns the modified list. |
| DIAGIN | Updates diagnostic arrays. |
| DIAGOUT | Prints diagnostic maps for meshes, showing hole points, fringe points, outer boundary points, and field points. |
| DWNPTRS | Sets up pointers that point "down" the mesh-boundary-surface hierarchy. |
| EQUAL | Checks to see if two character arrays are equal. |
| FDELTA | Calculates smallest distance between points in a mesh. Values are used to define how close two points from different meshes must be to be considered coincident. |
| FFRNGPTS | Finds fringe points surrounding hole points. |

| | |
|---|---|
| FNDNBRS | Finds the neighboring absolute indices of a given mesh index. |
| GETMESH | Inputs $MESH NAMELIST records, sets up connection information, and checks for consistency. |
| GETSURF | Inputs $SURFACE NAMELIST records, sets up surface-boundary connections, and checks for consistency. |
| HBIBLANK | Blanks out hole points in a mesh. |
| INBOX | Checks to see if a point is within a rectangular region defined by the minimum and maximum coordinates of a mesh. |
| INITDAF | Initializes direct access files. |
| INPUT | Top-level input module of PEGSUS. Inputs meshes and NAMELIST records, and performs syntax and consistency checks. |
| INTDAT | Computes interpolation coefficients for trilinear interpolation. Performs secondary function of "stencil-jumping" during search for interpolation elements. |
| INTPTS | Returns an array containing indices of either fringe or outer boundary points, depending on the value of an argument. |
| ISINSIDE | Returns .TRUE. if a mesh point is inside a surface, .FALSE. otherwise. |
| ITOJKL | Calculates J,K,L indices from a given absolute Index I. |
| JKLTOI | Calculates an absolute Index I from given J,K,L indices. |
| MESHCHK | Checks consistency of meshes and NAMELIST information. |
| MESHCHK2 | Checks consistency of $MESH NAMELIST records. |
| MESHDEF | Sets default values for $MESH NAMELIST records. |
| MESHINP | Inputs all $MESH NAMELIST data, performs consistency checking, and prints mesh informtion. |

| | |
|---|---|
| MESHNO | Returns a mesh number, given a mesh name. |
| MESHSTAT | Prints mesh statistics. |
| NEWTON | Performs trilinear interpolation using Newton's method. |
| NORMAL | Calculates normal vectors at all points on a surface. The surface is assumed to be defined in a right-hand coordinate system. A left-hand system will result in surface normals facing inward, which will cause holes to be generated incorrectly. |
| NUMSPTS | Returns the number of points on a surface. |
| NWIBLANK | Sets all elements of an array corresponding to an input index list to a given value. |
| NWIDIAG | Same as NWIBLANK, but uses characters instead of integers. |
| OBIBLANK | Blanks out outer boundary points in each mesh. Only outer boundary points that belong to designated interpolation boundaries are blanked. |
| OPTSRCH | Calculates optimum search range for hole location. If PEGSUS is run again on the same mesh configuration, the optimum search range can be input by the user. |
| PARAMCHK | Checks PARAMETER statement and warns user if parameters are not large enough. |
| PEGSUS30 | Main calling routine. |
| PRCHOLES | Processes hole boundaries in each mesh, i.e., finds interpolation elements, interpolation coefficients, boundary points, and pointers between boundary points and interpolation elements. |
| PRCOUTER | Processes outer boundaries. Almost identical to PRCHOLES, but works on outer boundary points rather than hole boundary points. |
| PREPROC | Edits NAMELIST input to allow embedded comments and indentation. |

| | |
|---|---|
| READMESH | Reads in separate meshes, performs scaling, translation, and saves each mesh in separate direct access files. |
| ROTATE | Rotates meshes about defined point in space. |
| SCALM | Scales meshes. |
| SRPINIT2 | Provides initial guesses for hole and outer boundary search routines. |
| STRIP | Removes comments from NAMELIST records and moves indented records left to Column 2. |
| SURFDEF | Defines $SURFACE NAMELIST default values. |
| SURFGEN | Creates direct access files corresponding to surfaces. |
| SYNCHK | Checks syntax of NAMELIST input. |
| TRANS | Translates meshes. |
| TRILIN | Finds an initial stencil reference point to begin search for interpolation stencils. Calls INTDAT to perform interpolation. Returns .TRUE. if interpolation succeeds, .FALSE. otherwise. |
| TSTCOPTS | Tests to see if a coincident point is a hole or boundary point. |
| TSTCRNRS | Tests to see if any corner of an interpolation element is a hole or boundary point. |
| WRITEOUT | Creates interpolation file from direct access files. |
| WRTINFO | Writes out information file. |
| XMSPECS | Modifies IBLANK arrays so blanked hole boundary and outer boundary points are blanked in the same way as hole points. Modifies IBC array so elements are relative, and not absolute, indices. Calculates IISPTR and IESPTR pointers. |

# APPENDIX E
# DIRECT ACCESS FILES

The following are stored as direct access files during the execution of PEGSUS 3.0.

Mesh Coordinates

| | |
|---|---|
| Unit | 12 |
| Record Format | X(MLEN),Y(MLEN),Z(MLEN) |
| Index | Mesh ID number |

Surface Definitions

| | |
|---|---|
| Unit | 13 |
| Record Format | XS(MSLEN),YS(MSLEN),ZS(MSLEN), VNX(MSLEN),VNY(MSLEN),VNZ(MSLEN),ISURF(MSLEN) |
| Index | Surface ID number |

Blanked Points

| | |
|---|---|
| Unit | 14 |
| Record Format | IBLANK(MLEN) |
| Index | Mesh ID number |

Interpolation Stencil Reference Points

| | |
|---|---|
| Unit | 15 |
| Record Format | JI(MSLEN),KI(MSLEN),LI(MSLEN) |
| Index | Mesh ID number |

Interpolation Coefficients

| | |
|---|---|
| Unit | 16 |
| Record Format | DXINT(MSLEN),DYINT(MSLEN),DZINT(MSLEN) |
| Index | Mesh ID number |

Boundary Points

| | |
|---|---|
| Unit | 17 |
| Record Format | IBC(MLEN) |
| Index | Mesh ID number |

IBC Arrays

| | |
|---|---|
| Unit | 18 |
| Record Format | JB(MSLEN),KB(MSLEN),LB(MSLEN) |
| Index | Mesh ID number |

Diagnostic Arrays

| | |
|---|---|
| Unit | 19 |
| Record Format | IDIAG(MLEN) (character array) |
| Index | Mesh ID number |

IBPLOT Arrays

| | |
|---|---|
| Unit | 20 |
| Record Format | IBPLOT (MLEN) |
| Index | Mesh ID number |

## APPENDIX F
## SAMPLE CRAY® JCL

Most users of chimera will (or should) run PEGSUS and XMER3D on a CRAY®. The job control program for the example of Figs. 20 through 23 is reproduced in this appendix.

```
JOB,JN=B30276A,T=30,MFL.
ACCOUNT,AC=55040906,US=B30276.
*
*  FETCH PREPROCESSOR, PEGSUS 3.0, AND MESH FILES
*  (PREPROCESSOR REPLACES PARAMETER STATEMENTS IN MASTER
*  PEGSUS 3.0 FILE WITH USER-DEFINED PARAMETER STATEMENT)
*
FETCH,DN=PREPROC,DF=CB,^
  TEXT='DSN=B09825.PEGSUS30.FORT(PREPROC),DISP=SHR'.
FETCH,DN=PEGSUS,DF=CB,^
  TEXT='DSN=B30276.PEGSUS30.FORT(PEGSUSP1),DISP=SHR'.
FETCH,DN=GRID1,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM3(CAVILG1),DISP=SHR'.
FETCH,DN=GRID2,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM3(CAVILG2),DISP=SHR'.
FETCH,DN=GRID3,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM3(CAVILG3),DISP=SHR'.
FETCH,DN=GRID4,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM3(CAVILG4),DISP=SHR'.
FETCH,DN=GRID5,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM3(CAVILG5),DISP=SHR'.
FETCH,DN=GRID6,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM3(CAVILG6),DISP=SHR'.
FETCH,DN=GRID7,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM5(STOREMA),DISP=SHR'.
FETCH,DN=GRID8,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM5(STINGMA),DISP=SHR'.
*
ACCESS,DN=DVSD,ID=SYSTEM,OWN=SYSTEM.
*
*  ASSIGN UNIT NUMBERS TO PEGSUS 3.0 AND PROCESSED PEGSUS
*
ASSIGN,DN=PEGSUS,A=FT35.
ASSIGN,DN=PEGSUS1,A=FT36.
```

```
*
*  COMPILE AND RUN PREPROCESSOR, WITH PEGSUS 3.0 TEXT FILE
*  AS INPUT
*
CFT,I=PREPROC,L=0.
LDR.
REWIND,DN=PEGSUS1.
*
*  PEGSUS1 CONSISTS OF PEGSUS 3.0 WITH MODIFIED PARAMETER
*  STATEMENTS
*
*  ASSIGN UNIT NUMBERS TO GRID, COMPOSITE MESH,
*  INTERPOLATION, PLOT3D, INFORMATION, MAP, AND RESTART FILES
*
ASSIGN,DN=GRID,A=FT11.
ASSIGN,DN=COMP,A=FT01.
ASSIGN,DN=INTRP,A=FT02.
ASSIGN,DN=BPLOT,A=FT10.
ASSIGN,DN=INFO,A=FT09.
ASSIGN,DN=MAPS,A=FT08.
ASSIGN,DN=RFIN,A=FT03.
ASSIGN,DN=RFOUT,A=FT04.
*
*  COPY ALL OF THE GRID FILES INTO ONE FILE FOR INPUT TO
*  PEGSUS 3.0
*
COPYR,I=GRID1,O=GRID,NR=3.
COPYR,I=GRID2,O=GRID,NR=3.
COPYR,I=GRID3,O=GRID,NR=3.
COPYR,I=GRID4,O=GRID,NR=3.
COPYR,I=GRID5,O=GRID,NR=3.
COPYR,I=GRID6,O=GRID,NR=3.
COPYR,I=GRID7,O=GRID,NR=3.
COPYR,I=GRID8,O=GRID,NR=3.
REWIND,DN=GRID.
*
*  COMPILE, LINK, AND RUN MODIFIED PEGSUS 3.0
*
CFT,I=PEGSUS1,ON=F,L=0.
LDR,SET=INDEF.
```

```
*
*  DISPOSE COMPOSITE MESH, INTERPOLATION FILE, PLOT3D IBLANK
*  FILE, INFORMATION FILE, AND MAPS FILE
*
DISPOSE,DN=COMP,DC=ST,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM5(COMPMA),DISP=SHR'.
DISPOSE,DN=INTRP,DC=ST,DF=TR,^

  TEXT='DSN=B30276.CAVITY.GEOM5(INTRMA),DISP=SHR'.
DISPOSE,DN=BPLOT,DC=ST,DF=TR,^
  TEXT='DSN=B30276.CAVITY.GEOM5(IBLNKMA),DISP=SHR'.

DISPOSE,DN=INFO,DC=ST,DF=CB,^
  TEXT='DSN=B30276.PEGOUT.DATA(INFOMA7),DISP=SHR'.
DISPOSE,DN=MAPS,DC=ST,DF=CB,^
  TEXT='DSN=B30276.PEGOUT.DATA(MAPSMA7),DISP=SHR'.
*
EXIT.
DUMPJOB.
DEBUG.
/EOF
  <  PARAMETER STATEMENT TO BE INSERTED INTO MASTER PEGSUS 3.0  >
   PARAMETER(MLEN  =50800, MSLEN  =10000, MDIM  =  10, NBDIM  =  20,
   &      NSDIM  =  60, NSEARCH  =  5)
  <PEGSUS 3.0 INPUT, FIG. 23 >
   .                               .
   .                               .
   .                               .
```

## APPENDIX G
## XMER3D INTERPOLATION FILE

PEGSUS 3.0 outputs an interpolation file for use with the XMER3D flow solver. The interpolation file, depicted schematically in this appendix consists of (1) a list of boundary points in each mesh, (2) a list of mesh elements in each mesh that provide interpolated information to the boundary points, and (3) pointers from the boundary points to the mesh elements. XMER3D uses the interpolation file to update boundary points in the composite mesh when computing a flow-field solution.

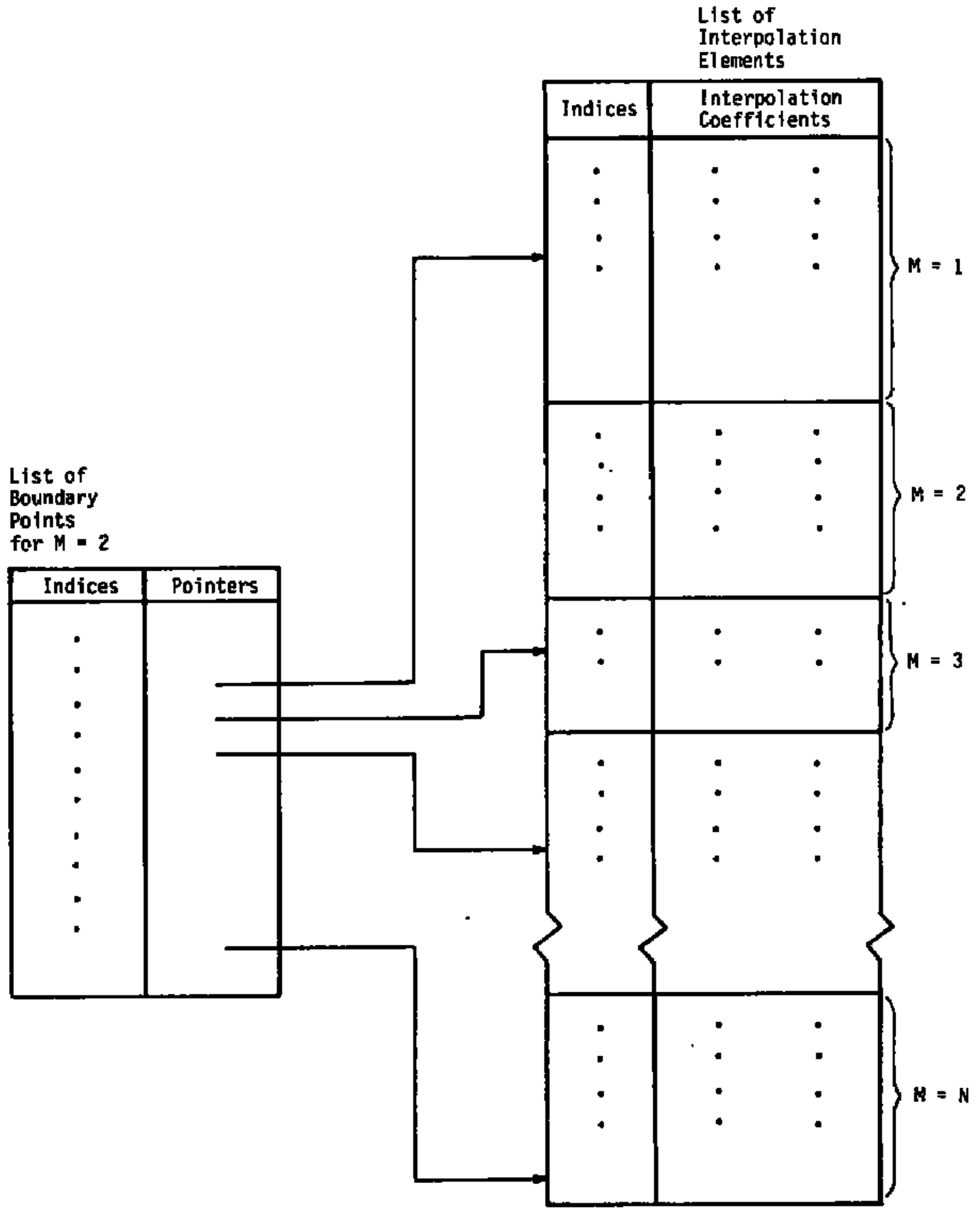

List of Interpolation Elements
Indices
Interpolation Coefficients
M = 1
M = 2
M = 3
M = N
List of Boundary Points for M = 2
Indices
Pointers

Figure G-1. Interpolation file data structure.